

msdn magazine



.NET in the Browser
with Blazor.....10, 48



Best-of-Breed UI Components for the Desktop, Web and Your Mobile World

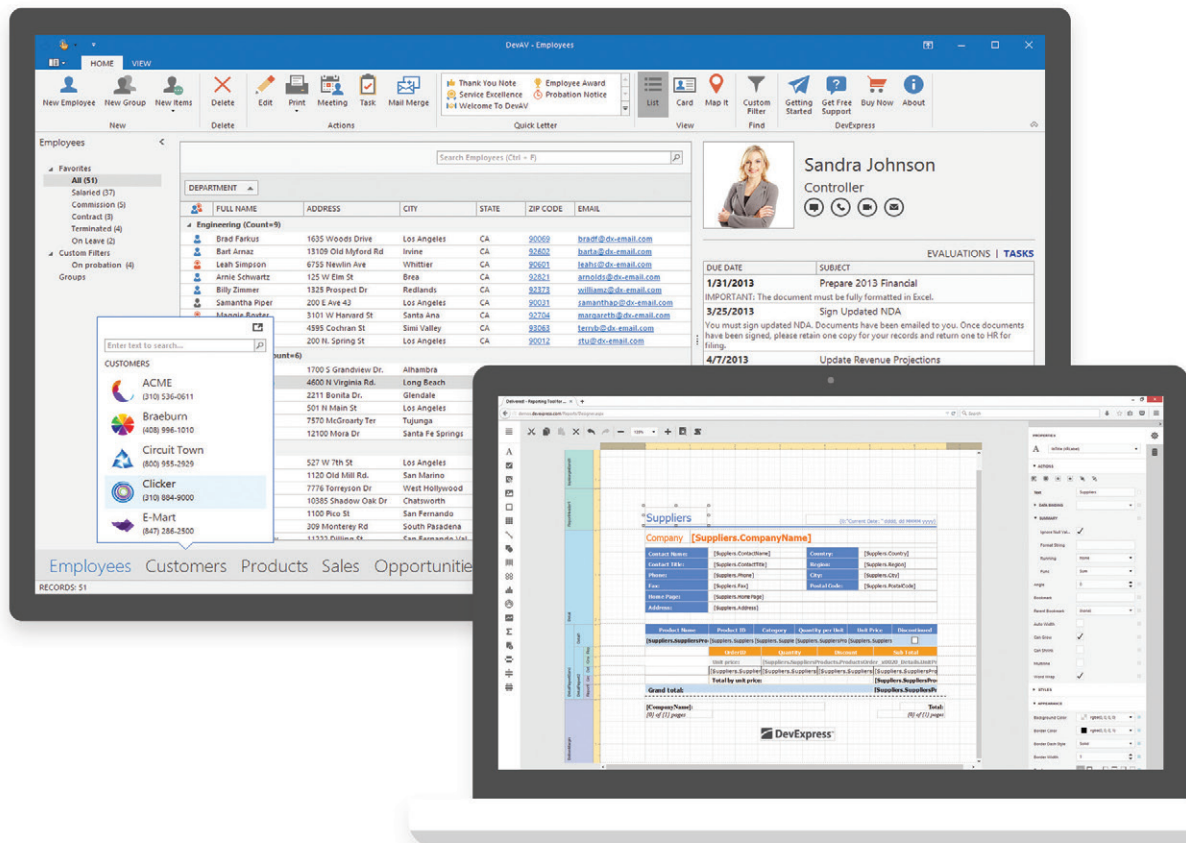
Free 30-Day Trial at
devexpress.com/trial





Your Next Great App Starts Here

From apps that replicate the look and feel of Microsoft Office® to high-powered data mining and decision support systems for your enterprise, DevExpress UI components for desktops, the web and mobile world will help you build your best, without limits or compromise.



Experience the DevExpress Difference
Download Your Free 30-Day Trial Today
devexpress.com/trial

All trademarks or registered trademarks are property of their respective owners.

msdn

magazine



.NET in the Browser
with Blazor.....10, 48

C# in the Browser with Blazor
Jonathan Miller 10

Managing Event Delivery with
Azure Event Grid
David Barkol 14

Architect Blockchain Applications
as Microservices
Stefano Tempesta 24

Xamarin Productivity and Platform
Improvements for Visual Studio 2017
Alessandro Del Sole 36

COLUMNS

THE WORKING PROGRAMMER

How To Be MEAN:
Routing Angular
Ted Neward, page 6

CUTTING EDGE

Never Mind JavaScript,
Here's Blazor
Dino Esposito, page 48

DON'T GET ME STARTED

Time of the Season
David S. Platt, page 56

Indigo.Design

A unified platform for Visual Design, UX Prototyping, Code Generation & App Development



Design to Match Your Brand

Create best-in-class UI designs using the expressive Indigo Design System with Sketch UI Kits. With 50+ components, 30+ UI patterns and complete app scenarios, getting started is a breeze.



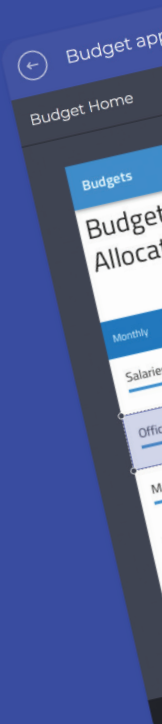
Share, Collaborate & Test

Import Sketch files into the cloud and share user flows and interactions on any device. Record & playback usability studies and get real-time reporting & analytics to see how users interact with your designs.



Runnable Code with 1-Click!

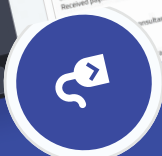
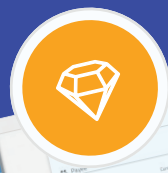
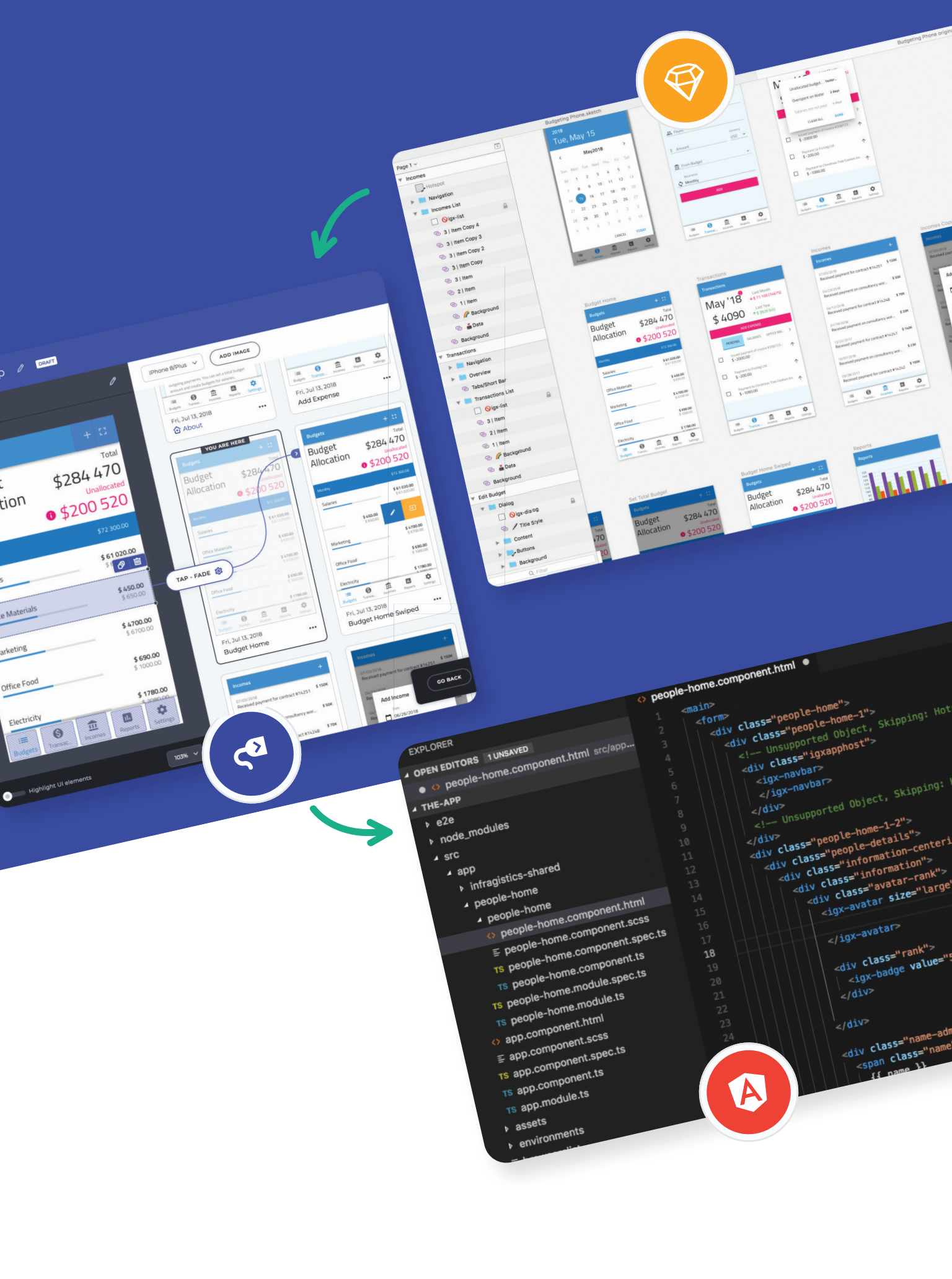
Everything you craft in Sketch from the design system matches to our Ignite UI for Angular components. With the click of a button, generate high-quality HTML, CSS, and Angular code from your design with no compromise.



Get it today for only \$99/month: indigo.design



To speak with sales or request a product demo with a solutions consultant call 1.800.231.8588



```
EXPLORER 1 UNSAVED
├── OPEN EDITORS
├── THE-APP
│   ├── e2e
│   ├── node_modules
│   └── src
│       ├── app
│       │   ├── infragistics-shared
│       │   ├── people-home
│       │   └── people-home
│       │       ├── people-home.component.html
│       │       ├── people-home.component.scss
│       │       ├── people-home.component.spec.ts
│       │       ├── people-home.module.spec.ts
│       │       ├── people-home.module.ts
│       │       ├── app.component.html
│       │       ├── app.component.scss
│       │       ├── app.component.spec.ts
│       │       ├── app.component.ts
│       │       ├── app.module.ts
│       │       ├── assets
│       │       └── environments
└── people-home.component.html
    1 <main>
    2 <form>
    3 <div class="people-home">
    4 <div class="people-home-1">
    5 <!-- Unsupported Object, Skipping: Hot
    6 <div class="igxapphost">
    7 <igx-navbar>
    8 </igx-navbar>
    9 </div>
    10 <!-- Unsupported Object, Skipping:
    11 <div class="people-home-1-2">
    12 <div class="people-details">
    13 <div class="information-center">
    14 <div class="avatar-rank">
    15 <igx-avatar size="large">
    16 </igx-avatar>
    17 </div>
    18 <div class="rank">
    19 <igx-badge value="
    20 </div>
    21 </div>
    22 </div>
    23 <div class="name-ad
    24 <span class="name
    25 {if name }>
```

General Manager Jeff Sandquist

Director Dan Fernandez

Editorial Director Jennifer Mashkowski mmeditor@microsoft.com

Site Manager Kent Sharkey

Editorial Director, Enterprise Computing Group Scott Bekker

Editor in Chief Michael Desmond

Features Editor Sharon Terdeman

Group Managing Editor Wendy Hernandez

Senior Contributing Editor Dr. James McCaffrey

Contributing Editors Dino Esposito, Frank La Vigne, Julie Lerman, Mark Michaelis, Ted Neward, David S. Platt

Vice President, Art and Brand Design Scott Shultz

Art Director Joshua Gould



Chief Revenue Officer
Dan LaBianca

ART STAFF

Creative Director Jeffrey Langkau

Associate Creative Director Scott Rovin

Art Director Michele Singh

Art Director Chris Main

Senior Graphic Designer Alan Tao

Senior Web Designer Martin Peace

PRODUCTION STAFF

Print Production Manager Peter B. Weller

Print Production Coordinator Lee Alexander

ADVERTISING AND SALES

Chief Revenue Officer Dan LaBianca

Regional Sales Manager Christopher Kourtoglou

Advertising Sales Associate Tanya Egenolf

ONLINE/DIGITAL MEDIA

Vice President, Digital Strategy Becky Nagel

Senior Site Producer, News Kurt Mackie

Senior Site Producer Gladys Rama

Site Producer, News David Ramel

Director, Site Administration Shane Lee

Front-End Developer Anya Smolinski

Junior Front-End Developer Casey Rysavy

Office Manager & Site Assoc. James Bowling

LEAD SERVICES

Senior Director, Audience Development & Data Procurement Annette Levee

Director, Audience Development & Lead Generation Marketing Irene Fincher

Director, Client Services & Webinar Production Tracy Cook

Director, Lead Generation Marketing Eric Yoshizuru

Senior Program Manager, Client Services & Webinar Production Chris Flack

Project Manager, Lead Generation Marketing Mahal Ramos

ENTERPRISE COMPUTING GROUP EVENTS

Vice President, Events Brent Sutton

Senior Director, Operations Sara Ross

Senior Director, Event Marketing Mallory Bastionell

Senior Manager, Events Danielle Potts

Senior Marketing Coordinator, Events Michelle Cheng



Chief Executive Officer
Rajeev Kapur

Chief Financial Officer
Craig Rucker

Chief Technology Officer
Erik A. Lindgren

Executive Vice President
Michael J. Valenti

Chairman of the Board
Jeffrey S. Klein

ID STATEMENT MSDN Magazine (ISSN 1528-4859) is published 13 times a year, monthly with a special issue in November by 1105 Media, Inc., 6300 Canoga Avenue, Suite 1150, Woodland Hills, CA 91367. Periodicals postage paid at Woodland Hills, CA 91367 and at additional mailing offices. Annual subscription rates payable in US funds are: U.S. \$35.00, International \$60.00. Annual digital subscription rates payable in U.S. funds are: U.S. \$25.00, International \$25.00. Single copies/back issues: U.S. \$10, all others \$12. Send orders with payment to: MSDN Magazine, P.O. Box 3167, Carol Stream, IL 60132, email MSDNmag@1105service.com or call 866-293-3194 or 847-513-6011 for U.S. & Canada; 00-1-847-513-6011 for International, fax 847-763-9564. POSTMASTER: Send address changes to MSDN Magazine, P.O. Box 2166, Skokie, IL 60076. Canada Publications Mail Agreement No: 40612608. Return Undeliverable Canadian Addresses to Circulation Dept. or XPO Returns: P.O. Box 201, Richmond Hill, ON L4B 4R5, Canada.

COPYRIGHT STATEMENT © Copyright 2018 by 1105 Media, Inc. All rights reserved. Printed in the U.S.A. Reproductions in whole or part prohibited except by written permission. Mail requests to "Permissions Editor," c/o MSDN Magazine, 4 Venture, Suite 150, Irvine, CA 92618.

LEGAL DISCLAIMER The information in this magazine has not undergone any formal testing by 1105 Media, Inc. and is distributed without any warranty expressed or implied. Implementation or use of any information contained herein is the reader's sole responsibility. While the information has been reviewed for accuracy, there is no guarantee that the same or similar results may be achieved in all environments. Technical inaccuracies may result from printing errors and/or new developments in the industry.

CORPORATE ADDRESS 1105 Media, Inc.
6300 Canoga Avenue, Suite 1150, Woodland Hills 91367
www.1105media.com

MEDIA KITS Direct your Media Kit requests to Chief Revenue Officer Dan LaBianca, 972-687-6702 (phone), 972-687-6799 (fax), dlabianca@1105media.com

REPRINTS For single article reprints (in minimum quantities of 250-500), e-prints, plaques and posters contact: PARS International
Phone: 212-221-9595
E-mail: 1105reprints@parsintl.com
www.1105Reprints.com

LIST RENTAL This publication's subscriber list, as well as other lists from 1105 Media, Inc., is available for rental. For more information, please contact our list manager, Jane Long, Merit Direct.
Phone: (913) 685-1301;
Email: jlong@meritdirect.com;
Web: www.meritdirect.com/1105

Reaching the Staff

Staff may be reached via e-mail, telephone, fax, or mail. E-mail: To e-mail any member of the staff, please use the following form: FirstInitialLastName@1105media.com
Irvine Office (weekdays, 9:00 a.m. – 5:00 p.m. PT)
Telephone 949-265-1520; Fax 949-265-1528
4 Venture, Suite 150, Irvine, CA 92618
Corporate Office (weekdays, 8:30 a.m. – 5:30 p.m. PT)
Telephone 818-814-5200; Fax 818-734-1522
9201 Oakdale Avenue, Suite 101, Chatsworth, CA 91311
The opinions expressed within the articles and other contents herein do not necessarily express those of the publisher.





Introducing LEADTOOLS Cloud Services

LEADTOOLS Cloud Services is a high-powered and scalable Web API that gives developers a hassle-free interface for implementing OCR, Document Conversion, MICR, and Barcode into any application.

Get Started Today

GET 50 FREE PAGES WHEN YOU SIGN UP

[SERVICES.LEADTOOLS.COM](https://services.leadtools.com)



.NET in the Browser

In February, Microsoft Senior Engineer Steve Sanderson announced that the ASP.NET team had taken formal ownership of Blazor—an experimental Web UI framework he created that's based on C#, Razor, and HTML and runs in the browser. The announcement set off a bit of a firestorm. After all, the project aims to do what earlier efforts like Silverlight could not: Enable developers to write .NET-based Web apps for the browser without a plug-in.

The secret behind Blazor is WebAssembly, a standard, binary instruction format that provides a portable target for compiling high-level languages like C++. WebAssembly is supported by all mainline browser families—desktop and mobile—released since mid-2017. Blazor leverages work from the Xamarin Mono project, enabling a .NET runtime for the Web. Code compiled to WebAssembly can run in any browser at native speeds.

Microsoft Principal Program Manager Daniel Roth has been elbow deep in Blazor going back to the February announcement, and helped vet two articles in this month's issue focused on it: Jonathan Miller's "C# in the Browser with Blazor" and Dino Esposito's Cutting Edge column, "Never Mind JavaScript, Here's Blazor." I asked Roth about his thoughts on Blazor and the promise it holds for .NET developers.

"With WebAssembly, for the first time, we have the opportunity to enable full-stack Web development with .NET, where you can use the .NET languages, frameworks, and tools to fully utilize the browser in an open and standardized way," Roth says. "We're seeing interest in Blazor for writing a wide variety of applications, from line-of-business apps to health care apps to games."

Since the February launch of the Blazor project, the ASP.NET team has been releasing updates every one to two months, with version 0.5.0 shipped in July. Roth says the most recent spin does some pretty cool stuff.

"Our latest Blazor release explored running Blazor in a separate process from the rendering process. For example, you might run Blazor in Web Worker, in an Electron app, or even server-side on .NET Core, and then handle all the UI updates and event handling over a cross-process channel."

Roth offers the example of Blazor running on the server-side and using SignalR to connect with clients. The approach eliminates the need to download a full .NET runtime to the browser.

"The nice thing about this model is that it's completely symmetric with running Blazor in the browser, but all the existing .NET libraries and tools just work, and you don't have to pay the cost of downloading a full .NET runtime into the browser."

Since the February launch of the Blazor project, the ASP.NET team has been releasing updates every one to two months, with version 0.5.0 shipped in July.

There's plenty of work ahead, says Roth. The WebAssembly-based .NET runtime needs improvement—smaller download sizes, better runtime performance and support for .NET Standard 2.0 are all priorities. The IL linker needs refinement to scrub more unused code from apps, and the team plans to explore support for Ahead-of-Time compilation of .NET code. And there's more.

"Besides the runtime, we need to complete the Blazor component model to support scenarios like template-based components, forms and validation, and handling more event types. We'd also love to have a tooling story for Blazor apps that can run cross-platform in Visual Studio Code," says Roth.

Blazor remains very much an experiment-in-progress, but one that could redefine Web application development. This much is certain: We'll be keeping an eye on this promising effort in the months and years to come.

Visit us at msdn.microsoft.com/magazine. Questions, comments or suggestions for *MSDN Magazine*? Send them to the editor: mmeditor@microsoft.com.

© 2018 Microsoft Corporation. All rights reserved.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, you are not permitted to reproduce, store, or introduce into a retrieval system *MSDN Magazine* or any part of *MSDN Magazine*. If you have purchased or have otherwise properly acquired a copy of *MSDN Magazine* in paper format, you are permitted to physically transfer this paper copy in unmodified form. Otherwise, you are not permitted to transmit copies of *MSDN Magazine* (or any part of *MSDN Magazine*) in any form or by any means without the express written permission of Microsoft Corporation.

A listing of Microsoft Corporation trademarks can be found at microsoft.com/library/toolbar/3.0/trademarks/en-us.mspx. Other trademarks or trade names mentioned herein are the property of their respective owners.

MSDN Magazine is published by 1105 Media, Inc. 1105 Media, Inc. is an independent company not affiliated with Microsoft Corporation. Microsoft Corporation is solely responsible for the editorial contents of this magazine. The recommendations and technical guidelines in *MSDN Magazine* are based on specific environments and configurations. These recommendations or guidelines may not apply to dissimilar configurations. Microsoft Corporation does not make any representation or warranty, express or implied, with respect to any code or other information herein and disclaims any liability whatsoever for any use of such code or other information. *MSDN Magazine*, MSDN and Microsoft logos are used by 1105 Media, Inc. under license from owner.

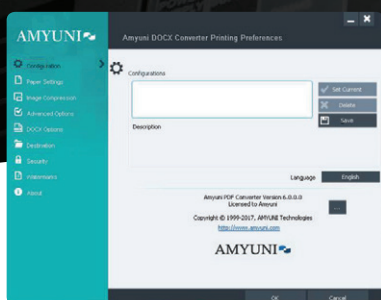


DOCXCONVERTER
For Windows

Free Demo at DOCXConverter.com

Amyuni DOCX Converter for Windows

Convert any document, including PDF documents, into DOCX format.
Enable editing of documents using Microsoft Word or other Office products.



A standalone desktop version, a server product for automated processing or an SDK for integration into third party applications.

Create

Create naturally editable DOCX documents with paragraph formatting and reflow of text

Convert

Convert images and graphics of multiple formats into DOCX shapes

OCR

Use OCR technology to convert non-editable text into real text

Extract

Extract headers and footers from source document and save them as DOCX headers and footers

Open

Open PDF documents with the integrated PDF viewer and quickly resave them to DOCX format

Configure

Configure the way the fonts are embedded into the DOCX file for optimal formatting

A virtual printer driver available for Windows 7 to Windows 10 and Windows Server 2008 to 2016

Powered by Amyuni Technologies:

Developers of the Amyuni PDF Converter and Amyuni PDF Creator products integrated into hundreds of applications and installed on millions of desktops and servers worldwide.

www.docxconverter.com

All trademarks are property of their respective owners. © Amyuni Technologies Inc. All rights reserved.



How To Be MEAN: Routing Angular

Welcome back again, MEANers.

Up to this point, despite all we've accomplished, everything has essentially been done entirely within the scope of one "page." While this does make sense for some single-page applications (SPAs), users of Web apps, even the most sophisticated Web apps (or the most sophisticated users) generally follow some of the established principles of the Web, like accessibility via a URL. That is to say, I should be able to "jump" to certain parts of the application by simply entering the appropriate URL into the browser, or bookmark a page while I'm on it, and so on. The ability to navigate "inside" the application is one of the hallmarks that distinguishes the Web app from the desktop or mobile app, and it's an important feature that should be supported.

In an older, more traditional Web application, this is handled by the very nature of the traditional ASP.NET (or Java servlets or Ruby-on-Rails or PHP or ...) application: Everything is a separate and distinct Web page that's manufactured on the server, then sent to the client for rendering. Within an SPA, however, most (if not all) of the rendering is done entirely client-side, and you only go back to the server when you need data or have to invoke some behavior that needs to remain tucked away on the server (such as modifying data in a shared database, or perhaps invoking a separate Web service hiding away behind the firewall on the user's behalf).

Thus, within most SPA frameworks—like Angular—a different mechanism is required in order to provide the kind of "scoping" or "segmenting" that page boundaries provide. In essence, you need some kind of tool or mechanism to change "pages" within the SPA, essentially ripping out whatever components are currently being displayed and replacing them with a different set of components, so that to the user, it looks like you changed pages, but without having to do the HTTP round-trip that navigating to a new page normally entails. Within Angular, that mechanism is called "routing," and it's the responsibility of the Angular Router. (And you, of course.)

Routing Fundamentals

To better grasp how routing works, let's assume a standard master-detail style of application: The first page will display a list of items in which I'm interested (such as speakers at a conference) by some sort of summarization (such as by last name and first name). When a user wants to "drill down" into a more detailed display of a single item, they'll click on that item in the list and I'll bring up a more detailed view. In Angular terms, this means I want to have two components to work with: a `SpeakerListComponent`, to

display the speakers by name, and a `SpeakerComponent`, to display the full details of that speaker. This is pretty standard master-detail stuff, and more importantly, it's the central staple of countless business applications. Naturally, there're other (perhaps better) ways of building a business UI, but this serves to get the core point across—I need to figure out how to route from the `SpeakerListComponent` to a given `SpeakerComponent`, and pass the selected speaker in while I'm at it.

A natural place to start with routing is with the collection of routes themselves. It's typical to want the homepage of the app to be the "master" view (the list of speakers), so let's start with that. As routing is usually something that's application-wide, or at least module-wide, typically routes are defined in the `app.module.ts` file, by constructing a `Routes` object imported from the "@angular/routing" module:

```
const appRoutes: Routes = [
  { path: 'speakers', component: SpeakerListComponent }
];
```

Note that the routes look essentially the same as they'd look in other Web technologies, like ASP.NET MVC or even Ruby-on-Rails. Routes are, at heart, a mapping of a URL path (without the leading slash) to a component that should be displayed. So, in this particular case, when a user navigates to "http://localhost:4200/speakers," they'll be rewarded with the list of speakers at the conference.

Come to think of it, I'd like the "/" path to redirect to the "speakers" route, so that coming to the "homepage" would automatically redirect to the list of speakers, so let's add that:

```
const appRoutes: Routes = [
  { path: 'speakers', component: SpeakerListComponent },
  { path: '', redirectTo: '/speakers', pathMatch: 'full' }
];
```

Of course, the other thing that's often needed is some kind of "Oops!" page to be shown if the user goes to a URL that doesn't exist on your site, so you also need a "wildcard" route that will display the `PageNotFoundComponent` you'll have the intern build over the summer:

```
const appRoutes: Routes = [
  { path: 'speakers', component: SpeakerListComponent },
  { path: '', redirectTo: '/speakers', pathMatch: 'full' },
  { path: '**', component: PageNotFoundComponent }
];
```

Note that the wildcard route doesn't really look any different, aside from the "*" path, but it's deliberately put at the end of the `Routes` table. This is by design, because routes are evaluated top-down, with the first match "winning." Thus, if the wildcard route were at the top, you'd always be showing off the intern's summer work, no matter whether the user navigated to a correct route or not.



ArcGIS for Developers

A complete mapping and analytics platform

The ArcGIS for Developers program offers a full suite of geospatial developer tools and resources. This program includes the ArcGIS Runtime SDK for .NET which enables developers to add map visualization and geospatial capabilities to native apps built using .NET. This empowers app users to do all things spatial—from simple map display and directions to editing and sharing of geographic data to advanced visual analysis. The ArcGIS Runtime SDK for .NET includes WPF, UWP, iOS, and Android APIs that enable .NET developers to build native mapping apps for Windows, iOS, and Android devices.

ArcGIS Runtime goes beyond basic mapping—it combines the power of the ArcGIS Platform with your data to deliver engaging, interactive, and high-quality mapping experiences how and where you need them. ArcGIS Runtime offers the following benefits to help you build and deploy native applications with powerful spatial capabilities:

High Speed Performance

ArcGIS Runtime is built on a C++ native core and takes advantage of GPU and other device resources to optimize display and interaction with both maps and data.

Work Offline

Use local maps and data to support offline workflows. ArcGIS Runtime supports local access to a wide variety of vector and raster data formats, including GeoPackage, Shapefile, and GeoTIFF. In addition, the ArcGIS Platform provides convenient tools for packaging and delivering maps and their data for offline use.

Advanced Cartography

Create and use beautiful, detailed, interactive symbols to optimize cartographic display, improve map readability, and communicate priorities in both 2D and 3D experiences.

Use the ArcGIS Platform

The ArcGIS Platform provides powerful tools, applications, and services to create, curate, stylize, host and secure your geographic data, maps, and workflows. The ArcGIS Platform also includes ready-to-use content and services you can combine with your data. ArcGIS Runtime works seamlessly to utilize and collaborate with all parts of the ArcGIS Platform to provide a comprehensive mapping solution.



Cross-Platform and Cross-Device

ArcGIS Runtime SDK for .NET integrates with the cross-platform capabilities of the .NET Platform. .NET developers can use ArcGIS Runtime to support mapping experiences and geospatial workflows in a single, shared codebase for native client applications that target Windows, iOS, and Android devices.

Integrated with Visual Studio

ArcGIS Runtime for .NET easily integrates with Visual Studio via NuGet and includes a set of application templates to get started. The SDK includes extensive guide and API reference documentation as well as source code for samples, example apps, and a toolkit to supplement native mapping components and workflows.

Get started at No Cost

Sign up for the ArcGIS Developer Program for free and get access to the ArcGIS Runtime SDK for .NET (and other ArcGIS APIs and SDKs): <https://developers.arcgis.com/sign-up>. Start building powerful mapping apps today!

About Esri

Esri, the global market leader in geographic information system (GIS) software, offers the most powerful mapping and spatial analytics technology available.

For more information, visit →

developers.arcgis.com

Figure 1 Asking for an `ActivatedRoute` in the Constructor

```
@Component({
  selector: 'app-speaker',
  templateUrl: './speaker.component.html',
  styleUrls: ['./speaker.component.css']
})
export class SpeakerComponent implements OnInit {
  @Input() speaker : Speaker

  constructor(private speakerService: SpeakerService,
    private route: ActivatedRoute) {
  }

  ngOnInit() {
    const speakerId = this.route.snapshot.params['id'];
    this.speaker = this.speakerService.getSpeakerById(speakerId);
  }
}
```

Last, I need to set up a route to display an individual speaker, and the typical way to do that is to give each speaker a page/route that makes use of some sort of unique identifier associated with them, something like `/speaker/1` for the speaker with ID 1. Setting up that route will look familiar to anyone with any familiarity with ASP.NET MVC or Rails, again, in that you use a colon-prefixed parameter name as a placeholder for the actual value passed in:

```
const appRoutes: Routes = [
  { path: 'speaker/:id', component: SpeakerComponent },
  { path: 'speakers', component: SpeakerListComponent },
  { path: '', redirectTo: '/speakers', pathMatch: 'full' },
  { path: '**', component: PageNotFoundComponent }
];
```

The only real question left on that subject is how the `SpeakerComponent` obtains the `“id”` parameter; therein lies an interesting tale.

ActivatedRoutes

When routing kicks in and brings a component to the screen, an `ActivatedRoute` object contains information about the route used, including the parameters to the route (such as the `“id”` used earlier). Like most things in Angular, an `ActivatedRoute` is an injectable object, so you can ask for it in the constructor of the `SpeakerComponent` to know which speaker to display, as shown in **Figure 1**.

The `ActivatedRoute` is heavily wrapped in observable entities, which is a bit more than I want to get into here, so suffice it to say that obtaining a `“snapshot”` of the route is the easiest way to get hold of the parameters passed in; from there I ask for the `“id”` parameter, and the `“1”` in `/speaker/1` is handed back to me for use with the `SpeakerService`.

The `ActivatedRoute` can operate via a number of different URL parts, by the way, in case you were wondering if the route can be mapped by URL parts or fragments or even query parameters. The short answer is yes, the `ActivatedRoute` can give you access to any part of the URL that you might want to parameterize, and the long answer is, of course, check the Angular documentation for all the details. In fact, routes can actually incorporate arbitrary data as part of the route, and resolve that data just prior to the route’s activation, but that’s a little far afield of what I have room to discuss this time around.

Router Outlets and Links

What’s missing still? Two parts: choosing where in the UI the router gets to do its magic, and linking between the master list and the

detail components. Both happen in template files, rather than the TypeScript code.

First, the most common place for the router to appear is usually in the root of the application itself; that is to say, the application component will most often define where the router’s various components should appear. In fact, usually the application component will have the router’s `“space”` surrounded by a header component above it, a footer component below it and so on. The router’s `“space”` is defined by the `“router-outlet”` tag, and it’s almost always empty—inside of it is where the routed components will appear, so my `app.component.html` will look like this:

```
div style="text-align:center">
  <h1>
    Welcome to {{ title }}!
  </h1>
</div>

<h2>Welcome to our conference</h2>
<router-outlet></router-outlet>

<h6>No speakers are actually going to appear, so...</h6>
```

Certainly, the conference Web site could do with a makeover, but I’ll leave that for a Web designer. The key here is that the `“router-outlet”` tag pair will be replaced by either the `SpeakerListComponent` or the `SpeakerComponent`, depending on which route is used.

The other thing that must be done is to put links in the `SpeakerListComponent`, so users can click on the speaker’s name and be taken to their details page. The easiest way to do this is to simply provide standard-issue hrefs in anchor tags in the `SpeakerListComponent`’s template, like so:

```
<div>
  <div *ngFor="let speaker of speakers">
    <a href="/speaker/{{speaker.id}}">
      {{speaker.firstName}} {{speaker.lastName}}</a>
    </div>
  </div>
```

And with that, we have a basic master-detail application.

Wrapping Up

There’s a great deal more to discover about routing—it’s a far more complex subject than I can completely cover here. For example, you can also specify an arbitrary (static) `“data”` parameter when defining the routes collection, which can then be picked up in the activated route object, or you can define what Angular calls a `“resolve guard”`, which can be used to do certain processing (such as retrieve the data for the selected speaker) while the UI is still being constructed. As always, the Angular documentation has copious details for those who are interested to learn more.

For now, however, we have our master-detail approach working, and it’s time for us to part ways for the month. In the next episode, we’ll talk about how to use Angular’s support for automated tests to test this sucker. In the meantime, happy coding! ■

TED NEWARD is a Seattle-based polytechnology consultant, speaker, and mentor, currently working as the director of Engineering and Developer Relations at Smartsheet.com. He’s written a ton of articles, authored and co-authored a dozen books, and speaks all over the world. Reach him at ted@tedneward.com or read his blog at blogs.tedneward.com.

THANKS to the following technical expert:
Garvice Eakins (Smartsheet.com)



From Desktops to Web and Mobile Your Next Great App Starts Here

Experience the DevExpress difference and see why your peers consistently vote our products #1. With our Universal Subscription, you will build your best, see complex software with greater clarity, increase your productivity and create stunning applications for Windows, Web and your Mobile world.



DevExpress Universal ships with 500+ UI controls.
It also includes our royalty-free reporting and dashboard platform.

WIN ASP MVC WPF UWP JS

Download your free 30-day trial today.
devexpress.com/try

All trademarks or registered trademarks are property of their respective owners.

C# in the Browser with Blazor

Jonathan C. Miller

Blazor is the new Microsoft experimental framework that brings C# into any browser without a plug-in. It holds the promise of modern single-page applications, combined with the ability to use C# and its vast base-class library. Blazor takes C# development to a new level. It's the final piece necessary to make the language a full-stack development tool. It will have all the power of the popular JavaScript frameworks, but based on the familiar languages, APIs and tooling of the Microsoft .NET Framework.

If you come from a traditional Microsoft background and are familiar with ASP.NET Web Forms or Model-View-Controller (MVC), getting up to speed on Blazor is surprisingly easy, especially when compared to the mountain a Microsoft developer must climb to gain the equivalent knowledge in a JavaScript framework such as Angular or React.

It's important to understand that Blazor runs completely inside the browser. Once a Blazor app is compiled, it's essentially a set of files that gets loaded into the browser and runs. Unlike traditional ASP.NET applications, there's no need for anything special on the back end to serve it. A Blazor site can be served by any Web server on any platform. As for clients, any browser that supports the WebAssembly standard supports Blazor. That includes all the major browsers shipping today.

When running in the browser, an application isn't really all that useful without access to external data and services. Just like standard JavaScript single-page applications, Blazor apps access Web services

using HTTP REST APIs. Those APIs can be created using Microsoft tools, such as Web API, or any technology that can present an HTTP REST endpoint. In this article, I'm going to demonstrate the ability of a Blazor app to call freely available Web services on the Web.

Getting Set Up

The first step is getting the current build of Blazor installed. As of this writing, Blazor is still an unsupported experimental framework. That means you shouldn't use it in production. The Blazor install is very low-impact, but you may not want to install it on your everyday work machine. Consider installing it on another machine or a virtual environment. The main requirements are Visual Studio 2017 with ASP.NET and Web development workload installed, the .NET Core SDK, and the Blazor Language extension. Please review the "Get Started" steps on blazor.net. The Blazor team is advancing quite rapidly, and sometimes you need specific versions of Visual Studio or .NET Core to use the current experimental version.

Creating a New Blazor App

I'm going to start by creating the sample Blazor application and then modify it to call some Web services. First, let's create a new ASP.NET Core Web Application in Visual Studio.

Next, choose a Blazor application and click OK. If you don't see Blazor in the list of choices, you may be missing the Blazor Language Services Extension.

Figure 1 Getting the ZIP Code Information

```
try
{
    errorMessage = "";
    var zipresultStr = await Http.GetStringAsync($"http://api.zippopotam.us/US/{zip}");
    zipresultStr = zipresultStr.Replace("place name", "city").Replace(
        "state abbreviation", "stateabbr");
    ziplookup = JsonUtil.Deserialize<Models.ZipLookup>(zipresultStr);
}
catch
{
    errorMessage = "Invalid zip code";
    return;
}
```

This article discusses:

- Creating a new Blazor app
- Calling external Web services
- Publishing the application

Technologies discussed:

C#, Blazor, ASP.NET Core

Code download available at:

msdn.com/magazine/0918magcode

The default Blazor application that's created is far from an empty canvas. It includes a basic Web site that utilizes Bootstrap. There are a few sample pages that make it easy to get up and running and start experimenting with Blazor right away. The Fetch data tab displays some dummy weather data from a JSON file embedded into the default application.

How Is This Even Possible?

The concept of C# running in a browser has been a dream since the inception of .NET Silverlight. It worked very well for line-of-business applications, but the fact that it required a plug-in and the emerging iOS model didn't allow browser plug-ins severely limited the future of Silverlight.

Figure 2 Deserializing the Raw JSON Data

```
public class CurrentConditions
{
    public CurrentConditions() { }
    public List<Weather> weather { get; set; }
    public Main main { get; set; }
    public String name { get; set; }
}
public class Weather
{
    public String main { get; set; }
    public String description { get; set; }
    public String icon { get; set; }
}
public class Main
{
    public decimal temp { get; set; }
    public decimal temp_min { get; set; }
    public decimal temp_max { get; set; }
}
```

Figure 3 Updating the Razor Script

```
<h1>
    @ziplookup.places[0].city, @ziplookup.places[0].stateabbr<br />
    @ConvertKtoF(currentcondition.main.temp, 1) &#176;F
</h1>
<h2>
    @currentcondition.weather[0].main 
</h2>
<h3>
    <span style="display:inline;color:red">HI
        @ConvertKtoF(currentcondition.main.temp_max, 0) &#176;F</span> /
    <span style="color:blue">LO @ConvertKtoF(
        currentcondition.main.temp_min, 0) &#176;F
    </span><br />
</h3>
```

Figure 4 Deserializing the JSON Data

```
public class Alert
{
    public String type { get; set; }
    public String title { get; set; }
    public Feature[] features { get; set; }
}
public class Feature
{
    public String type { get; set; }
    public PropertyInfo properties { get; set; }
}
public class PropertyInfo
{
    public String headline { get; set; }
    public String description { get; set; }
    public DateTime effective { get; set; }
    public DateTime expires { get; set; }
}
```

The magic that makes this all possible is a new standard called WebAssembly (WASM). WASM is a binary format that can be loaded and run directly in the browser and is currently supported by all the major browsers. The Mono team is working hard on a version of the .NET runtime that runs in WASM. Blazor itself is a framework that builds on top of that Mono runtime for WASM. When the project is compiled, it's compiled to a .NET assembly that gets loaded and executed by the Common Language Runtime (CLR) running inside the browser. It's important to understand that the .NET code running in the browser is running in the same JavaScript security sandbox. See Dino Esposito's Cutting Edge column in this issue for more on this topic.

Getting a Real Weather Forecast

The forecast is fake data loaded from a file embedded in the project. I'm going to replace the entire Fetch data page with real data from real Web services. First, I need to replace the HTML with something simple that prompts the user for a ZIP code. Inside the FetchData.cshtml, I replace the HTML code with the following:

```
<h1>Weather Forecast</h1>
<div class="input-group col-md-3">
    <input type="text" class="form-control" placeholder="Zip code"
        bind="@zip" maxlength="5" />
    <div class="input-group-append">
        <button class="btn btn-secondary" type="button"
            onclick="@GetWeather">Get Weather</button>
    </div>
</div>
<br /><span style="color:red">@errorMessage</span>
```

Notice the Razor syntax embedded in the script. The @ sign signals code and variables. The input tag captures the ZIP code and binds it to a variable called zip. The button tag has its onclick method bound to @GetWeather, which calls the GetWeather method in C# (not JavaScript). There's also a little @errorMessage that can be used if the user enters an invalid ZIP. These variables and methods are defined in the same FetchData.cshtml inside the @functions block:

```
@functions {
    String zip = String.Empty;
    String errorMessage = String.Empty;

    private async Task GetWeather()
    {
    }
}
```

Running the application now gives the user the ability to enter the ZIP code and click the Get Weather button. The GetWeather

Figure 5 Displaying Weather Alerts

```
<table class="table">
    <thead>
        <tr>
            <th>Date</th>
            <th>Alert</th>
        </tr>
    </thead>
    <tbody>
        @foreach (var alert in alerts.features)
        {
            <tr>
                <td>@alert.properties.effective.ToString("MM/dd/yyyy hh:mm")</td>
                <td>
                    <span style="font-weight:600">@alert.properties.headline</span><br />
                    <span>@alert.properties.description</span>
                </td>
            </tr>
        }
    </tbody>
</table>
```

method is empty, so nothing happens. In the next section, I'll add the code that will call the Web services and retrieve the current weather conditions.

Storing the Data Returned from the Web Service

Next, I need some local variables in my page to store the data returned from the Web service calls. The Razor pages will bind to these variables so the data can be displayed. These variables get added to the `@functions` block in the page, like so:

```
Models.CurrentConditions currentcondition;
Models.Alert alerts;
Models.ZipLookup ziplookup;
String imgUrl = "";
```

Adding the ZIP Code Web Service

Once the user clicks the Get Weather button, the ZIP code in the input box must be validated. The first public Web service is from `zippopotam.us`. When the `http://api.zippopotam.us/US/<zip>` API URL is called, it returns information about the specified ZIP code. The information needed from this Web service is the name of the city and state. This data will be displayed to the user in the forecast, and the state abbreviation will be used in subsequent Web service calls. The Web service code should look familiar because it uses the familiar `HttpClient` class.

Calling the ZIP Lookup Web Service The script in **Figure 1** downloads the ZIP code info from the API and places it into a local variable called `ziplookup`. I can use this variable in my Razor code to display the city name. The API will return an exception if the ZIP is invalid. If that happens, an error message is displayed.

Deserializing the ZIP Lookup Data In the previous code snippet, I'm retrieving data from the Web service and deserializing it into a `Models.ZipLookup` class. This is a class I've created to match the schema of the JSON data being returned:

```
public class ZipLookup
{
    public Place[] places { get; set; }
}
public class Place
{
    public String city { get; set; }
    public String stateabbr { get; set; }
}
```

Much more data is returned, but I've only created properties and classes for the data I want to use. The current implementation has issues with dealing with spaces in the JSON field names. As a temporary workaround, I'm using `String.Replace` to remove the spaces.

Displaying the City and State Now that the data has been downloaded and deserialized, I can display it in the Web page. The following code block displays the city and state abbreviation in the page:

```
<h1>
    @ziplookup.places[0].city, @ziplookup.places[0].stateabbr<br />
</h1>
```

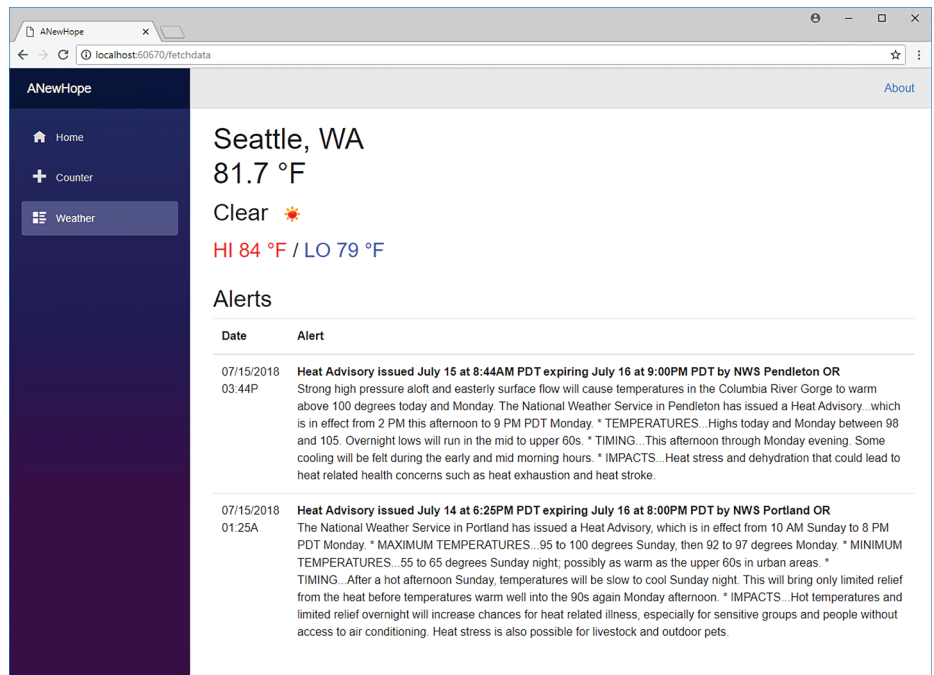


Figure 6 The Finished Application

Adding the Weather Conditions Web Service

The next Web service will retrieve the current conditions for the ZIP code from the `openweathermap.org` Web service. You'll need to create an account in order to receive a special key that's used when calling the Web service.

Calling the Current Conditions Web Service The call to get the current conditions works much like the previous Web service call. The exception here is the `apikey` parameter in the call. The `openweathermap.org` service requires a key to authenticate the caller:

```
currentcondition = await Http.GetJsonAsync<Models.CurrentConditions>(
    $"http://api.openweathermap.org/data/2.5/
    weather?zip={zip},us&appid={apikey}");
imgurl = $"http://openweathermap.org/img/w/{currentcondition.weather[0].icon}.png";
```

The result of the current conditions call is stored in a local variable called `currentcondition`. That result also passes the name of an icon to be displayed that corresponds to the current conditions. The name of the image is encoded into the `imgurl` variable so it can be displayed in the Web page.

Deserializing Current Conditions Data Once again, the raw JSON data needs to be deserialized into a class so it can be used, as shown in **Figure 2**. The class definition looks a little odd, but it's designed to match the schema of the JSON data being returned from the Web service. There's a lot more data being returned than what's shown here. Only the properties that are needed have to be implemented.

Converting the Temperatures The temperatures returned from the Web service are in kelvins, so the values need to be converted to degrees Fahrenheit and rounded. The current temperature will be rounded to the nearest tenth of a degree. The high and low temperatures will be rounded to the nearest whole degree.

```
private decimal ConvertKtoF(decimal kelvin, int decimals)
{
    return Math.Round(kelvin * 9 / 5 - 459.67M, decimals);
}
```



2018 Orlando

ROYAL PACIFIC RESORT AT UNIVERSAL ORLANDO

DECEMBER 2-7, 2018



The Ultimate Education Destination

Live! 360: A Unique Conference for the IT and Developer Community

- 6 FULL Days of Training Including Hands-On Labs & Workshops
- 6 Co-Located Conferences for 1 Low Price
- Customize Your Own Agenda from Hundreds of Sessions
- Expert Education and Training
- Knowledge Share and Networking



SEE THE FULL LIVE! 360
AGENDA INSIDE!



LIVE360EVENTS.COM

EVENT PARTNERS



SILVER SPONSORS



EXHIBITORS



SUPPORTED BY



PRODUCED BY





LIVE! 360 AGENDA-AT-A-GLANCE

Visual Studio LIVE!

DevOps in the Spotlight | Cloud, Containers and Microservices | App Dev Data | Developing New Experiences | Delivery and Deployment | .NET and More | Full Stack Web Development

ModernApps LIVE!

Presented in Partnership with: **Magenic**

SQL Server LIVE!

Business Intelligence | SQL Server Administration & Maintenance | SQL Server Features & Components | SQL Server for Developers

START TIME	END TIME	Visual Studio Live! Full Day Hands-On Labs: Sunday, December 2, 2018						SQL Server Live! Full Day Hands-On Lab: Sunday, December 2, 2018			
9:00 AM	6:00 PM	VSS01 Hands-On Lab: Develop an ASP.NET Core 2 and EF Core 2 App in a Day - Philip Japikse	VSS02 Hands-On Lab: From 0-60 in a Day with Xamarin and Xamarin.Forms - Roy Cornelissen & Marcel de Vries	VSS03 Hands-On Lab: Busy Developer's Workshop on VueJS - Ted Neward				SQ501 Hands-On Lab: Developer Dive into SQL Server - Leonard Label			
START TIME	END TIME	Visual Studio Live! Pre-Conference Workshops: Monday, December 3, 2018						SQL Server Live! Pre-Conference Workshops: Monday, December 3, 2018			
8:30 AM	5:30 PM	VSM01 Workshop: Web Development in 2018 - Chris Klug	VSM02 Workshop: Architect and Build a Modern ASP.NET App in the Azure Cloud with a full CI/CD Pipeline with VSTS - Brian Randall and Miguel Castro	VSM03 Workshop: Cross-Platform C# Using .NET Core and WebAssembly - Rockford Lhotka & Jason Bock				SQM01 Workshop: Planning SQL Server Solutions in a Physical and Cloudy World - Allan Hirt	SQM02 Workshop: How Data Science is Changing the Warehouse - Bradley Ball, Josh Luedeman, and Grant Fritchey		
6:30 PM	8:00 PM	Dine-A-Round Dinner @ Universal CityWalk - 6:30pm - Meet at Conference Registration Desk to walk over with the group						Dine-A-Round Dinner @ Universal CityWalk - 6:30pm - Meet at Conference Registration Desk to walk over with the group			
START TIME	END TIME	Visual Studio Live! Day 1: Tuesday, December 4, 2018						SQL Server Live! Day 1: Tuesday, December 4, 2018			
8:00 AM	9:00 AM	VISUAL STUDIO LIVE! & MODERN APPS LIVE! KEYNOTE: .NET Everywhere and for Everyone James Montemagno						SQL SERVER LIVE! & ARTIFICIAL INTELLIGENCE LIVE! KEYNOTE: To Be Announced James Montemagno			
9:15 AM	10:30 AM	VST01 ASP.NET Core 2 For Mere Mortals - Philip Japikse	VST02 Busy Developer's Guide to Kotlin - Ted Neward	VST03 Azure IoT - Laurent Bugnion	VST04 Modernizing Your Source Control: Migrating to Git from TFVC - Colin Dembovsky			SQT01 Availability Fundamentals for SQL Server - Allan Hirt	SQT02 What's New with Azure Data Factory - Josh Luedeman	SQT03 Azure IoT Introduction - Laurent Bugnion	
10:30 AM	11:00 AM	Networking Break • Visit the EXPO - Pacifica 7						Networking Break			
11:00 AM	12:00 PM	LIVE! 360 KEYNOTE: To Be Announced - Pacifica 6						LIVE! 360 KEYNOTE: To Be Announced - Pacifica 6			
12:00 PM	1:30 PM	Lunch • Visit the EXPO / Dessert Break • Visit the EXPO						Lunch • Visit the EXPO / Dessert Break • Visit the EXPO			
1:30 PM	1:50 PM	VST05 Fast Focus: Serverless Computing: Azure Functions and Xamarin in 20 minutes - Laurent Bugnion	VST06 Fast Focus: Getting Git - Jason Bock	VST07 Fast Focus: Xamarin Essentials - Cross-Platform APIs for Your Mobile Apps - James Montemagno	VST08 Fast Focus: Get Your Full .NET Code into .NET Standard - Rockford Lhotka			SQT04 Fast Focus: SQL Server Data Security and Privacy Features - Thomas LaRock	SQT05 Fast Focus: Common T-SQL and How to Fix Them - Amy Herold		
2:00 PM	2:20 PM	VST08 Fast Focus: Getting Started with ASP.NET Core 2.0 Razor Pages - Walt Ritscher	VST09 Fast Focus: Cross Platform Device Testing with xUnit - Oren Novotny	VST10 Fast Focus: What is this GDPR Thing? - Michael Lester				SQT06 Fast Focus: Graph DB Support in SQL Server - Karen Lopez	SQT07 Fast Focus: SQL Server Performance - Grant Fritchey		
2:20 PM	2:45 PM	Networking Break • Visit the EXPO - Pacifica 7						Networking Break			
2:45 PM	4:00 PM	VST11 Introduction to Webpack - Chris Klug	VST12 Xamarin.Forms Takes You Places! - Sam Basu	VST13 Using the Cloud to Power Your Cross-platform Applications - Laurent Bugnion	VST14 Writing Testable Code and Resolving Dependencies - Miguel Castro			SQT08 Data Protection and Privacy in the Database World - Grant Fritchey	SQT09 Introduction to SQL Server Essential Concepts - Bradley Ball	SQT10 Performance Tuning - Michael Lester	
4:15 PM	5:30 PM	VST15 Angular Application Testing Outside the Church of TDD - Chris Klug	VST16 Busy .NET Developer's Guide to Python - Ted Neward	VST17 HoloLens, Mixed Reality & VR Development with the Cloud - Nick Landry	VST18 Testing in Production Using Azure and Visual Studio Team Services (VSTS) - Colin Dembovsky			SQT11 Exploring Execution Plans - Grant Fritchey	SQT12 Everything You Need to Know About SQL Server Indexes - Janis Griffin	SQT13 Reading XE's Issues - Grant Fritchey	
5:30 PM	7:30 PM	Exhibitor Reception - Pacifica 7						Exhibitor Reception - Pacifica 7			
START TIME	END TIME	Visual Studio Live! Day 2: Wednesday, December 5, 2018						SQL Server Live! Day 2: Wednesday, December 5, 2018			
8:00 AM	9:15 AM	VSW01 The Whirlwind Tour of Authentication and Authorization with ASP.NET Core - Chris Klug	VSW02 Essential Tools for Xamarin Developers! - Sam Basu	VSW03 Introduction to Windows Containers and Docker - Marcel de Vries	VSW04 OWASP DevSlop: DevSecOps with VSTS & Azure - Tanya Janca			SQW01 Top Tips for Deploying AGs and FCIs On Premises or in the Cloud - Allan Hirt	SQW02 Level Up Your SQL Server Cloud Skills - David Klee	SQW03 Azure Building Cosms - Laurent Bugnion	
9:30 AM	10:45 AM	VSW05 Assembling the Web - A Tour of WebAssembly - Jason Bock	VSW06 Cross-Platform App Dev with Xamarin and CSLA .NET - Rockford Lhotka	VSW07 Microservices with AKS (Azure Kubernetes Service) - Vishwas Lefe	VSW08 To Be Announced			SQW04 What's New in the 2017 Query Store - Janis Griffin	SQW05 Redundant Devs & DBAs - Adaptive Query Processing and Automatic Tuning - Pinal Dave	SQW06 Power BI Done Right - Amy Herold	
10:45 AM	11:30 AM	Networking Break • Visit the EXPO - Pacifica 7						Networking Break			
11:30 AM	12:30 PM	LIVE! 360 KEYNOTE: Enterprise Transformation (And You Can Too) Donovan Brown, Principal DevOps Manager, Cloud Developer Advocacy Team, Microsoft						LIVE! 360 KEYNOTE: Enterprise Transformation (And You Can Too) Donovan Brown, Principal DevOps Manager, Cloud Developer Advocacy Team, Microsoft			
12:30 PM	2:00 PM	Birds-of-a-Feather Lunch / Dessert Break • Visit the EXPO						Birds-of-a-Feather Lunch / Dessert Break • Visit the EXPO			
2:00 PM	3:15 PM	VSW09 JavaScript without Webpack, Transpilers, or Frameworks - Ashley Grant	VSW10 DevOps for the SQL Server Database - Brian Randall	VSW11 Programming with Microsoft Flow - Walt Ritscher	VSW12 What's New in C# 7 - Phil Japikse			SQW07 Leveraging Data Value with Azure Data Catalog - Karen Lopez	SQW08 Upgrading to SQL Server 2017 - Thomas LaRock	SQW09 DevOps: Improving Your CI/CD Pipeline - Grant Fritchey	
3:15 PM	4:00 PM	Networking Break • Visit the EXPO • Expo Raffle @ 3:30 p.m. - Pacifica 7						Networking Break • Visit the EXPO • Expo Raffle @ 3:30 p.m. - Pacifica 7			
4:00 PM	5:15 PM	VSW13 Electron: Desktop Development for Web - Chris Woodruff	VSW14 Entity Framework for Enterprise Applications - Benjamin Day	VSW15 The Mystical World of I/O Bindings in Azure Functions - Ashley Grant	VSW16 Signing Your Code the Easy Way - Oren Novotny			SQW10 The New Rules of SQL Server Monitoring - Grant Fritchey	SQW11 Table Indexing - Denny Cherry	SQW12 SQL Server Optimal Configuration Learning Series - Grant Fritchey	
7:30 PM	9:00 PM	Live! 360 Dessert Luau - Wantilan Pavilion						Live! 360 Dessert Luau - Wantilan Pavilion			
START TIME	END TIME	Visual Studio Live! Day 3: Thursday, December 6, 2018						SQL Server Live! Day 3: Thursday, December 6, 2018			
8:00 AM	9:15 AM	VSH01 Encrypting the Web - Robert Boedigheimer	VSH02 Learning HTTP for a Better Mobile App Data Experience - Chris Woodruff	VSH03 How to Interview a Developer - Billy Hollis	VSH04 Creating a Release Pipeline with Team Services - Esteban Garcia			SQH01 Cosmos DB for SQL Server Pros - David Klee	SQH02 Performance in 60 Seconds - SQL Tricks Everybody MUST Know - Pinal Dave	SQH03 How Data is Stored in SQL - Laurent Bugnion	
9:30 AM	10:45 AM	VSH05 Advanced Fiddler Techniques - Robert Boedigheimer	VSH06 Making XAML Do Things You Didn't Realize It Could - Billy Hollis	VSH07 What Developers Want - Rabeb Othmani	VSH08 To Be Announced			SQH04 HiHol HiHol SQL Server on Linux, We Go! - Janis Griffin	SQH05 SQL Server 2016 Database Administration for the non-DBA - Denny Cherry	SQH06 What's New in SQL Server 2017 - Grant Fritchey	
11:00 AM	12:00 PM	VISUAL STUDIO LIVE! PANEL DISCUSSION: What Matters Most for the Future of Modern Apps: AI, Data, Security, or UX? Brian Randall (Moderator), Billy Hollis, Tanya Janca, Oren Novotny, & Rabeb Othmani						SQL SERVER LIVE! & ARTIFICIAL INTELLIGENCE LIVE! PANEL DISCUSSION: Keeping Pace with AI and Machine Learning While Maintaining Your Joy Andrew Brust (moderator); Thomas LaRock, Jen Stirrup, Jen Underwood, & Stacia Varga			
12:00 PM	1:00 PM	Lunch on the Lanai - Lanai / Pacifica 7						Lunch on the Lanai - Lanai / Pacifica 7			
1:00 PM	2:15 PM	VSH09 Get Func-y: Understanding Delegates in .NET - Jeremy Clark	VSH10 Non-Useless Unit Testing Entity Framework & ASP.NET MVC - Benjamin Day	VSH11 Modernizing the Enterprise Desktop Application - Oren Novotny	VSH12 DevOps: A Catalyst for Enterprise Agility - Heidi Araya & Esteban Garcia			SQH07 SQL Server Design Features: Contentious Issues - Karen Lopez	SQH08 Virtual SQL Servers, Actual Performance - David Klee	SQH09 Let's Pretend SQL Server is a Database - Amy Herold	
2:30 PM	3:45 PM	VSH13 IEnumerable, ISaveable, IDontGetIt: Understanding .NET Interfaces - Jeremy Clark	VSH14 When and Why You Should Consider Azure Cosmos DB - Eric Potter	VSH15 Pushing Left Like a Boss: Application Security Foundation - Tanya Janca	VSH16 Scrum, Kanban, or Scrumban? - Heidi Araya			SQH10 SQL Server Audit - Thomas LaRock	SQH11 Secrets of SQL Server - Database Worst Practices - Pinal Dave	SQH12 Reading XE's Issues - Grant Fritchey	
4:00 PM	5:00 PM	Next7 Live! 360 Networking Event						Next7 Live! 360 Networking Event			
START TIME	END TIME	Visual Studio Live! Post-Conference Workshops: Friday, December 7, 2018						SQL Server Live! Post-Conference Workshops: Friday, December 7, 2018			
8:00 AM	5:00 PM	VSP01 Workshop: UX Design for Developers: Basics of Principles and Process - Billy Hollis	VSP02 Workshop: Microservices with AKS (Managed Kubernetes) - Vishwas Lefe					SQF01 Workshop: Migrating Data and Databases to Microsoft Azure - Karen Lopez & Thomas LaRock	SQF02 Workshop: Data Due Diligence Strategy for BI, Analytics, and Data Science - Stacia Varga		

Speakers and sessions subject to change

ROYAL PACIFIC RESORT AT UNIVERSAL ORLANDO | DECEMBER 2-7



		Artificial Intelligence LIVE						TECHMENTOR					Office & SharePoint LIVE					
SQL Server Performance Tuning and Optimization		AI App Development	Analytics	Bots	Data Science and ML	Internet of Things	Deep Learning	Client and Endpoint Management	PowerShell and DevOps	Infrastructure	Soft Skills for IT Pros	Security	Cloud (Public/Hybrid/Private)	Managing Office 365	SharePoint/Office 365/Cloud	High-Value SharePoint Workloads	Developing for Office/SharePoint	Office 365 Identity and Security
18		Artificial Intelligence Live! Full Day Hands-On Lab: Sunday, Dec 2																
		AI501 Hands-On Lab: Build a Bot in a Day Using the Microsoft Bot Framework, Cognitive Services and Azure - Brian Randall																
2018		Artificial Intelligence Live! Pre-Conference Workshops: Mon, Dec 3																
AIM01 Workshop: So You Want to do Data Science, What Now? - Matt Winkler & Euan Garden		AIM02 Workshop: Moving from BI to AI: Artificial Intelligence Skills for Business Intelligence Professionals - Jen Stirrup																
Meet at Conference Registration Desk to walk over with the group																		
		Artificial Intelligence Live! Day 1: Tuesday, December 4, 2018																
		ARTIFICIAL INTELLIGENCE LIVE! & SQL SERVER LIVE! KEYNOTE: To Be Announced																
e Cosmos DB Part I - on to Cosmos DB Standard Label		AIT01 Make Your App See, Hear and Think with Cognitive Services - Roy Cornelissen																
		AIT02 Introduction to Azure Databricks - Andrew Brust																
• Visit the EXPO - Pacifica 7																		
To Be Announced - Pacifica 6																		
/ Dessert Break • Visit the EXPO																		
QL Coding Mistakes		AIT03 Fast Focus: Externalizing Conversational Context - Heather Downing																
Execution Plans		AIT05 Fast Focus: Supervised VS Unsupervised Machine Learning Approach - Leila Etadi																
• Visit the EXPO - Pacifica 7		AIT07 Speak to Me: Voice App Conversation Practices - Heather Downing																
ing Between the Lines - Diagnose Application Mindy Currutt		AIT09 Taking IoT to the Edge - Eric D. Boyd																
Reception - Pacifica 7																		
		Artificial Intelligence Live! Day 2: Wednesday, December 5, 2018																
e Cosmos DB Part II - os DB Applications Standard Label		AIW01 Building for Alexa with Web API - Heather Downing																
		AIW02 Deep Learning in Microsoft Azure: CNTK, CaffeOnSpark and Tensorflow - Jen Stirrup																
		AIW03 Practical Internet of Things for the Microsoft Developer - Eric D. Boyd																
• Visit the EXPO - Pacifica 7		AIW04 The Case for R in AI Applications - Jen Stirrup																
Donovan Brown, Principal DevOps Manager, Cloud Developer Advocacy Team, Microsoft																		
n / Dessert Break • Visit the EXPO																		
Design Techniques for Lower BI Visualizations Ginger Grant		AIW05 Building a Holographic Assistant with Bot Framework, LUIS & Mixed Reality - Nick Landry																
PO • Expo Raffle @ 3:30 p.m. - Pacifica 7		AIW06 Data Visualization Super Power - Reza Rad																
Server Internals and Configuration for Machine Visions - Ginger Grant		AIW07 ML for the .NET Developer - Matt Winkler & Euan Garden																
Luau - Wantilan Pavilion		AIW08 Intelligent Reports with Azure Cognitive Services and Power BI - Leila Etadi																
		Artificial Intelligence Live! Day 3: Thursday, December 6, 2018																
ow Modern is Your Warehouse? - Stacia Varga		AIH01 Managing the Conversation Flow within a Bot - Rabeb Othmani																
		AIH02 Scaling Machine Learning in Azure - Matt Winkler & Euan Garden																
ere Does R Fit Into I-Server Stack? - Stacia Varga		AIH03 Machine Learning the Easy Way: Azure Cognitive Services - Eric Potter																
		AIH04 Chest X-ray Image Analysis with Deep Learning - Vishwas Lele																
b Varga		ARTIFICIAL INTELLIGENCE LIVE! & SQL SERVER LIVE! PANEL DISCUSSION: Keeping Pace with AI and Machine Learning While Maintaining Your Day Job Andrew Brust (moderator); Thomas LaRock, Jen Stirrup, Jen Underwood, & Stacia Varga																
Lanai - Lanai / Pacifica 7																		
and This Never Happened: Coding Mistakes and How to Avoid Them - Amy Herold		AIH05 DevOps for AI - Matt Winkler & Euan Garden																
		AIH06 How to Avoid Building Bad Predictive Models - Jen Underwood																
World PowerShell Deep Learning - Amy Herold		AIH07 Getting Started with Deep Learning - Seth Juarez																
		AIH08 Innovations in Automating Analytics and Machine Learning - Jen Underwood																
60 Networking Event																		
2018		Artificial Intelligence Live! Post-Conference Workshop: Fri, Dec 7																
ence - Developing a and Beyond		AIW01 Workshop: Practical Artificial Intelligence for the Working Software Engineer - Seth Juarez																
		TechMentor Post-Conference Workshops: Friday, December 7, 2018																
		TMF01 Workshop: Storage Spaces Direct & Managing Windows Server with Project Honolulu - Dave Kawula																
		TMF02 Workshop: Hacking and Hardening Office 365 and Azure AD - Nestori Synnima																
		Office & SharePoint Live! Post-Conference Workshop: Fri, Dec 7																
		OSF01 Workshop: Developing Components for SharePoint and Teams - Paul Schaefflin																

REGISTER NOW AT [LIVE360EVENTS.COM/MSDN](https://live360events.com/msdn)



2018 Orlando

ROYAL PACIFIC RESORT
AT UNIVERSAL ORLANDO
DECEMBER 2-7



REGISTER
NOW

REGISTER BY
OCTOBER 5 AND
SAVE \$400!*

Use Promo Code: MSDN

* Restrictions Apply.

CONNECT WITH LIVE! 360

twitter.com/live360
@live360
facebook.com
Search "Live 360"
linkedin.com
Join the "Live! 360" group!
instagram.com
@live360_events

6 CO-LOCATED CONFERENCES, 1 GREAT PRICE!

Visual Studio LIVE!

EXPERT SOLUTIONS FOR .NET DEVELOPERS

Visual Studio Live! features unbiased and practical development training on the Microsoft Platform. Come join us and code in paradise!

SQL Server LIVE!

TRAINING FOR DBAs AND IT PROS

SQL Server Live! will leave you with the skills needed to drive your data to succeed, whether you are a DBA, developer, IT Pro, or Analyst.

TECHMENTOR

IN-DEPTH TRAINING FOR IT PROS

TechMentor is where IT training meets sunshine, with zero marketing speak on topics you need training on now, and solid coverage on what's around the corner.

NEW! IN 2018 Artificial Intelligence LIVE!

AI FOR DEVELOPERS AND DATA SCIENTISTS

Artificial Intelligence Live! is an innovative, new conference for current and aspiring developers, data scientists, and data engineers covering artificial intelligence (AI), machine learning, data science, Big Data analytics, IoT & streaming analytics, bots, and more.

Office & SharePoint LIVE!

ON-HOURLY, CLOUD & CROSS-PLATFORM TRAINING

Office & SharePoint Live! provides leading-edge knowledge and training to work through your most pressing projects and enable people to work from anywhere at any time.

ModernApps LIVE!

MOBILE, CROSS-DEVICE & CLOUD DEVELOPMENT

Modern Apps Live!, presented in partnership with Magenic, focuses on how to architect, design and build a complete Modern App from start to finish.



LIVE360EVENTS.COM



HIGHLIGHTED KEYNOTES AND FEATURED SPEAKERS

Live!360 brings you so many options to hear from our experts in our Keynotes and Sessions. Check out a few of our Highlighted Keynotes and Featured Speakers, bringing some of the best and brightest experts in the industry to our full conference agenda. View the full list of speakers at live360events.com



Live! 360 Keynote: Enterprise Transformation (And You Can Too)
Donovan Brown, Principal DevOps Manager, Cloud Developer Advocacy Team, Microsoft



.NET Everywhere and for Everyone
James Montemagno, Principal Program Manager, Mobile Developer Tools, Microsoft



OneDrive and M365 - Enabling and Secure and Seamless Collaboration for Untethered Workforce



Jason Moore, Partner Group Program Manager, Microsoft
Stephen Rose, Sr. Product Manager, OneDrive for Business, Microsoft

Visual Studio LIVE



Miguel Castro
President, Mervcorp LLC



Billy Hollis
XAML Singer



Tanya Janca
Senior Cloud Advocate, Microsoft



Rabeb Othmani
Developer Advocate, Neumo



Brian Randall
Partner, MCW Technologies

SQL Server LIVE



Pinal Dave
SQL Server Performance Tuning Expert



Thomas LaRock
Head Geek, SolarWinds



Leonard Lobel
CTO, Slack Technologies, Inc.



Karen Lopez
Data Evangelist, Space Enthusiast, & NASA Outbound



Jorge Segarra
Sr. Consultant, Premier Support for Developers, Microsoft

TECHMENTOR



Jessica Deen
Azure Advocate, Microsoft



Dana Enache
Azure Active Directory Product Group, Microsoft



Jeffery Hicks
PowerShell Sensei



Don Jones
Technical Curriculum Dir., Pluralsight



Bruce Mackenzie-Low
Master Consultant

Artificial Intelligence LIVE



Heather Downing
Coder, Entrepreneur, Passionate for the Possible



Seth Juarez
Cloud Developer Advocate, Microsoft



Vishwas Lele
CTO, Applied Information Sciences



Jen Stirrup
Principal, Data Relish Ltd



Matt Winkler
Group Program Manager, Machine Learning at Microsoft

Office & SharePoint LIVE



Neil Hodgkinson
Senior Program Manager, O365 CKP Team, Microsoft



Matthew McDermott
Founder and Director, Aptilion, Inc.



Agnes Molnar
Founder, Managing Consultant, Search Explained



Eric Shupps
Founder & President, BinaryWare



Aimee Triplet
Engineer, Fast Track Center, Microsoft

ModernApps LIVE



Jim Barret
Senior UX Designer, Magenic Technologies



Jason Bock
Practice Lead, Magenic Technologies



Allen Conway
Senior Consultant, Magenic Technologies



Brent Edwards
Principal Lead Consultant, Magenic Technologies



Rockford Lhotka
CTO, Magenic Technologies



REGISTER BY OCTOBER 5 AND SAVE \$400!*

Use Promo Code MSDN

*Restrictions apply. See website for details.

Displaying the Current Conditions The Razor script in Figure 3 will now be updated to include the current temperature, the high and low, a description of the current conditions, and an icon that corresponds to the current conditions. Notice how the temperatures are being passed to the ConvertKtoF function created earlier.

Adding National Weather Service Alerts

The final Web service will retrieve alerts from the National Weather Service (NWS) for the state the ZIP code is in.

Calling the NWS Web Service The following block of code retrieves all the current severe weather alerts from the NWS:

```
alerts = await Http.GetJsonAsync<Models.Alert>($"https://api.weather.gov/alerts/active/area/{ziplookup.places\[0\].stateabbr}");
```

The state abbreviation retrieved from the ziplookup Web service is used to filter the alerts to the same state.

Deserializing NWS Alerts Data The JSON data needs to be deserialized into a set of classes that aligns with the NWS JSON schema. The code snippet in Figure 4 holds the alerts returned from the NWS Web service.

Displaying the Severe Weather Alerts Now that the alerts have been retrieved, the code in Figure 5 is added to the Razor view to display them. It creates a table to display the alerts and uses @foreach to loop through and display each alert. The effective date, headline, and description are displayed for each alert. If there are no alerts for the state, the table will be empty.

Putting It All Together

The application is now complete, as shown in Figure 6. It runs entirely in the browser and calls three external Web services to display the current weather conditions and alerts.

Publishing the Blazor Application

Publishing a Blazor application is just as easy as publishing any other ASP.NET application. Doing this from Visual Studio will compile the application code and generate a complete Web application with all of the final HTML, CSS, scripts and Blazor binaries needed. A good starting place is to use the Publish to Folder option. All of the files of the application get put there. If you look inside the Dist folder, you'll find a standard index.htm page and an _framework folder. The _framework folder contains all of the compiled assemblies, as well as the Blazor and Mono runtime components.

Next Steps

Blazor is still an experimental framework, though the team is moving forward at a very fast rate of release. The Blazor roadmap details a full framework with routing, components, layouts and more. The Blazor project is being developed in the open on GitHub. Visit the blazor.net page to keep up-to-date and start experimenting with the current bits. ■

JONTHAN MILLER is a senior architect. He's been developing products on the Microsoft stack for a decade and programming on .NET since its inception. Waldman is a full-stack product developer with expertise in front-end technologies (Windows Forms, Windows Presentation Foundation, Silverlight, ASP.NET, AngularJS/Bootstrap), middleware (Windows services, Web API), and back ends (SQL server, Azure).

THANKS to the following technical experts who reviewed this article:
Dino Esposito (BaxEnergy), Daniel Roth (Microsoft)

msdnmagazine.com



Instantly Search Terabytes

dtSearch's **document filters** support:

- popular file types
- emails with multilevel attachments
- a wide variety of databases
- web data

Over 25 search options including:

- efficient multithreaded search
- **easy** **multicolor** **hit-highlighting**
- forensics options like credit card search

Developers:

- APIs for .NET, C++ and Java; ask about new cross-platform .NET Standard SDK with Xamarin and .NET Core
- SDKs for Windows, UWP, Linux, Mac, iOS in beta, Android in beta
- FAQs on faceted search, granular data classification, Azure and more

Visit [dtSearch.com](https://dtsearch.com) for

- hundreds of reviews and case studies
- fully-functional enterprise and developer evaluations

The Smart Choice for Text Retrieval[®]
since 1991

dtSearch.com 1-800-IT-FINDS

Managing Event Delivery with Azure Event Grid

David Barkol

Event Grid is a fully managed messaging service in Microsoft Azure that provides an innovative approach for the routing of events in the cloud and beyond. It has unlocked new and unique patterns for how event-driven solutions are designed with a powerful and flexible publish-subscribe model.

I provided an introduction to Azure Event Grid in the February 2018 issue (aka.ms/eventgridarticle) that explores the fundamentals of the service and how it can be used to publish and consume events in various ways. In this new article, I'll delve into how events are delivered and what the options are for retry policies, invalid events and events that aren't successfully delivered. Before I dive in, it will be helpful to go over how Event Grid works at a high level.

Some of the technology discussed in the article is in preview; all information is subject to change.

This article discusses:

- Event delivery and response codes
- Invalid and dead letter channels
- Dead lettering with Event Grid
- Azure Functions event handler
- Subscribing to, sending and inspecting events

Technologies discussed:

Microsoft Azure, Event Grid, Logic Apps, Functions

Code download available at:

bit.ly/2LGKjhN

A Brief Overview of Event Grid

With Azure Event Grid, event sources can originate from a growing list of services in Azure, such as Event Hubs and Media Services, or even from a custom application that's running on-premises or within another cloud provider or datacenter. These events are consumed and managed by Event Grid, which is responsible for the ingesting of the messages and for their distribution to each event subscription. Event handlers are used to take action on incoming events and can be services in Azure. A very popular scenario is one that leverages other serverless technologies such as Functions and Logic Apps. Together, these highly scalable and flexible solutions can be composed very quickly and affordably without the burden of managing any infrastructure.

An event handler can also be a simple WebHook, which means that just like an event source, it can reside anywhere as long as it supports HTTPS and can accept a POST request. This platform- and language-agnostic approach is one of the many options that make Event Grid a very special service that has only just begun to open up new solutions in the cloud. **Figure 1** illustrates how Event Grid is used to connect multiple sources and handlers. The list of services in Azure that integrate with Event Grid is constantly growing and only a subset is depicted in the diagram.

Event Delivery and Response Codes

Event Grid treats each event independently. This means that there isn't a guaranteed order for the events and, in some cases, an event can be delivered more than once. Therefore, it's the responsibility of the event handler to code defensively and be idempotent. Sending the

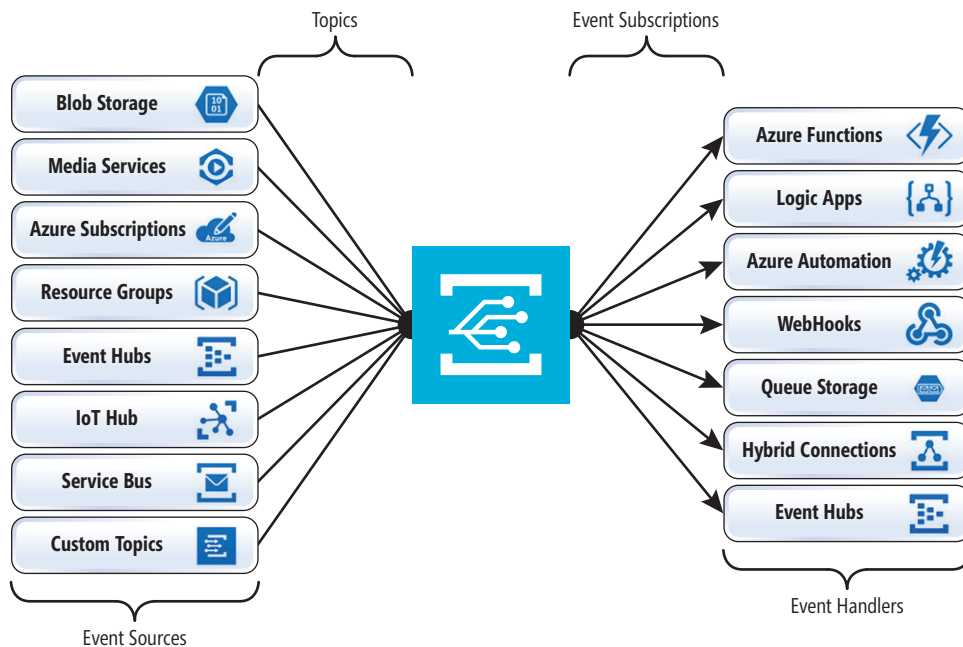


Figure 1 Azure Event Grid Overview

same event repeatedly should produce the same result. If the ordering of events is a requirement, this should be managed on either the compute side (within the logic of the event handler) or by using another service such as Service Bus or Event Hubs to preserve their order.

The HTTP response that's returned by the event handler to Event Grid will determine how it will proceed with the management of the event. The status codes 200 OK and 202 Accepted are considered to be an acknowledgement of a successfully delivered event.

Any of the following failure codes are indicative of a failed delivery attempt: 400 Bad Request, 401 Unauthorized, 404 Not Found, 408 Request Timeout, 414 URI Too Long, 500 Internal Server Error, 503 Service Unavailable and 504 Gateway Timeout. Depending on the failure code, Event Grid might retry sending the event to the endpoint. I'll dig into this in just a bit.

Retry Policies

If an acknowledgement isn't received or an error code is returned, another attempt will be made to send the event to the endpoint. A retry policy that employs an exponential backup is then put into place to attempt a final delivery of the event before it expires. By default, an event will expire after 24 hours of unsuccessful delivery. After the

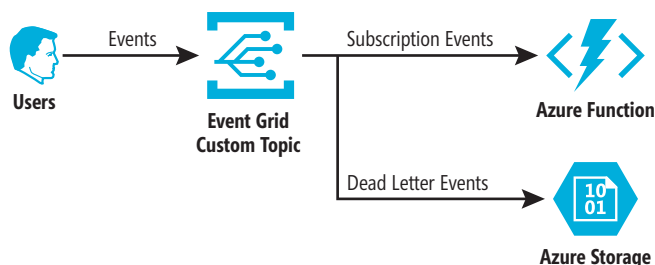


Figure 2 Dead Letter Events

first attempt, the delivery schedule will back off, using the following timeline: 10 seconds, 30 seconds, 1 minute, 5 minutes, 10 minutes, 30 minutes and 1 hour. After the first one-hour attempt, each subsequent request is made once per hour until the time-to-live (TTL) is reached.

A new feature in Event Grid allows you to configure the retry policy for an event subscription by setting two possible values:

Max delivery attempts is a configurable value that sets the maximum retry attempts for an event subscription. Its default value, and maximum allowed, is 30.

Event TTL is a value that corresponds to the time-to-live setting for an event. Its default value, as well as its maximum setting, is 1,440 minutes (24 hours).

Either of these properties can be

used to manage the retry policy at the event subscription level and can be useful when an event is interesting to you only for a short period of time, or you're intentionally throwing an error such as 503 to protect yourself during high load.

Invalid and Dead Letter Channels

One of the primary responsibilities of a messaging service is to ensure that a message is delivered properly. However, it can't guarantee that the receiver of that message will handle it successfully.

In some cases, the receiver of an event might reject the message if it doesn't meet certain expectations. This could come in the form of an invalid data type or payload, or an unauthorized message. When this happens, the message is typically moved to a designated location, often referred to as an *invalid message channel*.

Another common scenario is a message that's successfully sent but can't be processed due to an error on the receiving end. These are typically the 500-level status codes that indicate a service error or service unavailability. Usually, the messaging service will retry sending these messages until a threshold is met and the message is deemed undeliverable. A *dead letter channel* is used, similar to the invalid message channel, to store these messages along with any relevant metadata.

The desire in both scenarios is that there will be some utility or service that monitors the channels and will know what to do with the messages. This could surface in a scheduled report or perhaps another event-driven service, such as a Logic App, that can pick up the message and notify end users or other systems and services.

Dead Lettering with Event Grid

A highly requested feature for Azure Event Grid shortly after it was released was to provide a mechanism for capturing events that couldn't be delivered. This feature has recently come to fruition with the ability to configure the retry policies and dead letter

Figure 3 Creating a Resource Group, Storage Account and Container

```
# create resource group
az group create --name $rgname -l westus2

# create storage account
az storage account create \
  --name $storagename \
  --location westus2 \
  --resource-group $rgname \
  --sku Standard_LRS \
  --kind StorageV2 \
  --access-tier Hot

# create storage container
export AZURE_STORAGE_ACCOUNT=$storagename
export AZURE_STORAGE_ACCESS_KEY="$(az storage account keys list --
  account-name $storagename
  --resource-group $rgname --query "[0].value" --output tsv)"

az storage container create --name $containername
```

location for each event subscription. With this feature, I can now configure the dead letter channel to an Azure Storage Account that will capture the message and relevant details about its delivery. This same endpoint will serve as both a dead letter and invalid message channel. **Figure 2** illustrates how dead letter events are now supported in Azure Event Grid.

I want to call out a couple of important details before putting a sample together that uses both the retry policies and dead letter delivery.

Notice that the arrows coming from the Event Grid topic are going in the direction of the handlers. I wanted to point this out to reinforce the notion that Event Grid is a push-push model. Event handlers provide a webhook that's called by Event Grid when it's time to send an event. This important design decision emphasizes the event-driven nature of the service. Unlike other messaging services, there's no longer a need to resort to long-polling or hammer-polling techniques to check if a message is available. Instead, a handler can rely on this

model to be notified of new events, which is another reason why serverless technologies such as Functions and Logic Apps for event handlers are appealing. Let's put this into practice.

Setup

Now I'll walk you through the steps of setting up and configuring both the retry policies and a dead letter endpoint. In addition, it would be great to also inspect the dead letter events and react to them immediately after they become available.

Everything I'm going to set up in Azure will be done from the Azure Cloud Shell. This will ensure that you can do it from any machine and not be at the mercy of any particular tools or other dependencies. Details about Azure Cloud Shell can be found at bit.ly/2CsFtQB.

At the time of this writing, this feature is in preview, though it most likely will be released before or shortly after the article is published. While it's in preview, you'll have to install the Event Grid extension to use it in the command-line interface (CLI):

```
az extension add --name eventgrid
```

With that out of the way, I'm going to initialize a few local variables that will be used repeatedly throughout the exercise:

```
rgname=msndemo
topicname=<your-unique-grid-topic-name>
storagename=<your-unique-storage-account-name>
containername=deadletterevents
```

I'm going to create a resource group, storage account and a container that will be used to receive the dead letter events, as shown in **Figure 3**.

Next, I need the storage account's resource ID. This will be used, along with the container name, to define the dead letter endpoint:

```
# storage ID
storageid=$(az storage account show --name $storagename
  --resource-group $rgname --query id --output tsv)

# container ID for dead letter channel
containerid="$storageid/blobservices/default/containers/$containername"
```

Figure 4 The Azure Function Event Handler

```
using System.IO;
using System.Linq;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Azure.WebJobs;
using Microsoft.Azure.WebJobs.Extensions.Http;
using Microsoft.AspNetCore.Http;
using Microsoft.Azure.EventGrid.Models;
using Microsoft.Azure.WebJobs.Host;
using Newtonsoft.Json;

namespace WeWantTheFunc
{
    public static class SongRequestHandler
    {
        [FunctionName("SongRequestHandler")]
        public static IActionResult Run(
            [HttpTrigger(AuthorizationLevel.Function, "post", Route = null)]
            HttpRequest req, TraceWriter log)
        {
            // Get the body of the request
            var requestBody = new StreamReader(req.Body).ReadToEnd();

            // Check the header for the event type
            if (!req.Headers.TryGetValue("Aeg-Event-Type", out var headerValues))
                return new BadRequestObjectResult("Not a valid request");

            var eventTypeHeaderValue = headerValues.FirstOrDefault();
            if (eventTypeHeaderValue == "SubscriptionValidation")
            {
                // Validate the subscription
                var events = JsonConvert.DeserializeObject<EventGridEvent[]>(requestBody);
                dynamic data = events[0].Data;
                var validationCode = data["validationCode"];
                return new JsonResult(new
                {
                    validationResponse = validationCode
                });
            }
            else if (eventTypeHeaderValue == "Notification")
            {
                // Handle the song request
                log.Info(requestBody);
                var events = JsonConvert.DeserializeObject<EventGridEvent[]>(requestBody);

                // Reject the request if it does not
                // match the genre for the station.
                if (events[0].Subject != "genre/blues")
                    return new BadRequestObjectResult("Sorry, this is a Blues station");

                return new OkObjectResult("");
            }

            return new BadRequestObjectResult("Not a valid request");
        }
    }
}
```

File Format APIs

Open, Create, Convert, Print and Save files from your applications!

Try risk free – 30 day trial



Download a Free Trial at



<https://downloads.aspose.com>



Aspose.Words

Create, edit, convert or print Word documents (DOC, DOCX, RTF etc.) in your .NET, Java and Android applications.



Aspose.Cells

Develop high performance .NET, Java and Android applications to Create, Edit or Convert Excel worksheets (XLS, XLSX, ODS etc).



Aspose.Pdf

Manipulate PDF file formats (PDF, PDF/A, XPS etc.) using our native APIs for .NET, Java and Android platforms.



Aspose.Slides

Create, edit or convert PowerPoint presentations (PPT, PPTX, ODP etc.) in your .NET, Java and Android applications.



Aspose.Email

Create, Edit or Convert Outlook Email file formats (MSG, PST, EML etc.) and popular network protocols.



Aspose.BarCode

Generate or recognize barcodes (Code128, PDF417, Postnet etc.) using our native APIs for .NET and Java.



Aspose.Imaging

Deliver efficient applications to Create, Draw, Manipulate or Convert image file formats.



Aspose.Tasks

Develop high performance apps to Create, Edit or Convert Microsoft Project® document formats.

► Aspose.Diagram ► Aspose.Note ► Aspose.3D ► Aspose.CAD ► Aspose.HTML ► Aspose.GIS

Americas: +1 903 306 1676

EMEA: +44 141 628 8900
sales@asposeptyltd.com

Oceania: +61 2 8006 6987

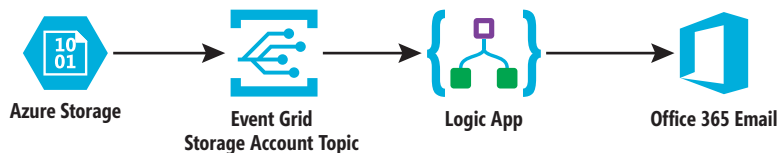


Figure 5 Blob Events Handled by Logic Apps

I'll be sending the events to a custom Event Grid topic. Let's create the topic and save the endpoint address and access key. These values will be leveraged later when publishing events:

```
# create custom topic
az eventgrid topic create -g $rgname --name $topicname -l westus2

# save topic endpoint
topickey=$(az eventgrid topic show --name $topicname -g $rgname --query "endpoint" --output tsv)

# save topic key
topickey=$(az eventgrid topic key list --name $topicname -g $rgname --query "key1" --output tsv)
```

Azure Functions Event Handler

In this scenario, I'll pretend that users are sending song requests to a fictitious radio station that specializes in Blues music. As requests come in, they're put onto a playlist for the station. However, if the genre of the song doesn't match the target music for the radio station, it will be rejected and placed on the dead letter channel.

The event handler is an Azure Function built on the version 2 runtime. It will inspect the Subject field of the event and approve or reject the song request accordingly. The code for the Azure Function is shown in **Figure 4**.

This function has two parts. The first checks to see if the event coming in is intended for validating the endpoint. If it's a validation request, the function returns the validation code to prove ownership and acceptance of incoming messages.

The second part of the function is for event notifications from Event Grid. By returning a bad request response (401), an explicit statement is made to no longer send the event to the handler. This brings me to the most important part of the process—creating an event subscription.

Subscribing to Events

With all the pieces in place, it's finally time to create an event subscription that demonstrates both the retry policies and dead lettering feature. An assumption is that the Azure Function in **Figure 4** has been deployed and is currently running in Azure. The command to create the subscription is as follows:

```
az eventgrid event-subscription create \
  --endpoint <your-azure-function-url> \
  --topic-name $topicname \
  -g $rgname \
  --name song-request-sub \
  --deadletter-endpoint $containerid \
  --max-delivery-attempts 2 \
  --event-ttl 1
```

Notice that both the max-delivery-attempts and event-ttl settings are included. It's not a requirement to include both settings, but they can be used together to configure the retry policies. I'm setting the maximum number of delivery attempts to two and the time to live to one minute. Another new argument called dead-letter-endpoint is initialized to the storage account container using the variable that was created earlier. It's time to send some events and see this working end-to-end.

Sending Events

From the CLI, I can copy a sample body that contains a song request. This request will actually contain an invalid music genre (Rock) that will be rejected by the event handler.

```
# copy the request body
body=$(eval echo "$(curl https://raw.githubusercontent.com/dbarkol/azure-event-grid-patterns/master/badsongrequest.json)")

# post the request to the custom topic endpoint
curl -X POST -H "aeg-sas-key: $topickey" -d "$body" $topicendpoint
```

The expectation is that a few minutes after the message is sent, it will end up in the storage account container that I configured as the

Figure 6 An Event Grid-Triggered Logic App

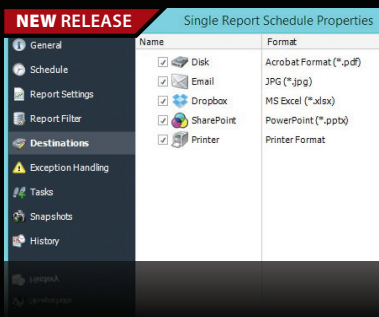


Help & Manual Professional | from \$586.04



Help and documentation for .NET and mobile applications.

- Powerful features in an easy, accessible and intuitive user interface
- As easy to use as a word processor, but with all the power of a true WYSIWYG XML editor
- Single source, multi-channel publishing with conditional and customized output features
- Output to responsive HTML, CHM, PDF, MS Word, ePub, Kindle or print
- Styles and Templates give you full design control

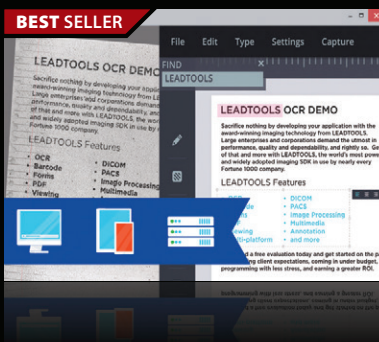


PBRS (Power BI Reports Scheduler) | from \$8,132.21



Schedules & delivers Power BI reports to unlimited recipients with one Power BI License.

- Exports reports to PDF, Excel, Excel Data, Word, PowerPoint, CSV, JPG, HTML, PNG and ePub
- Sends reports to email, printer, fax, folder, FTP, DropBox and SharePoint
- Uses database queries to automatically populate report filters, email addresses & body text
- Adds flexibility with custom calendars e.g. 4-4-5, holidays, "nth" day of the month, etc.
- Responds instantly by firing off reports when an event occurs e.g. database record is updated



LEADTOOLS Document Imaging SDKs V20 | from \$2,995.00 SRP



Add powerful document imaging functionality to desktop, tablet, mobile & web applications.

- Universal document viewer & conversion framework for PDF, Office, CAD, TIFF & more
- OCR, MICR, OMR, ICR and Forms Recognition supporting structured & unstructured forms
- PDF SDK with text edit, hyperlinks, bookmarks, digital signature, forms, metadata
- Barcode Detect, Read, Write for UPC, EAN, Code 39, Code 128, QR, Data Matrix, PDF417
- Zero-footprint HTML5/JavaScript UI Controls & Web Services



DevExpress DXperience 18.1 | from \$1,439.99



The complete range of DevExpress .NET controls and libraries for all major Microsoft platforms.

- WinForms - New TreeMap control, Chart series types and Unbound Data Source
- WPF - New Wizard control and Data Grid scrollbar annotations
- ASP.NET - New Vertical Grid control, additional Themes, Rich Editor Spell Checking and more
- Windows 10 Apps - New Hamburger Sub Menus, Splash Screen and Context Toolbar controls
- CodeRush - New debug visualizer expression map and code analysis diagnostics

dead letter channel. You can use a tool like Azure Storage Explorer to see the new blob that's created for each dead-lettered message. However, that isn't very exciting. As an alternative, I want to be notified when a new message is added to the dead letter channel.

Inspecting Dead Letter Events

Because the dead letter event is just a new blob created in a container, Event Grid can be used to kick off another workflow that's triggered when the file is created. **Figure 5** demonstrates the workflow for how an event can originate from a storage account and ultimately be received by a Logic App and send an email.

The only thing I'll need to put this together is a Logic App that begins with an Event Grid trigger. The first three steps in the application comprise the following actions and trigger:

Event Grid trigger is configured for the storage account. It uses two filters—one for the event type (Microsoft.Storage.BlobCreated) and the second for the container using the prefix filter option. Both of these filters ensure that the app is only invoked when a new file is created within the dedicated dead letter container.

Initialize variable is the next action that retrieves the URL from the body of the Event Grid message. Its value is the following expression:

```
triggerBody()?['data']['url']
```

Get blob content using path is the action that will read the contents of the blob. I format the value of the blob path using this string manipulation expression to remove the portion that isn't needed:

```
replace(variables('DeadLetterUrl'),
'https://<storage-account-name>.blob.core.windows.net', '')
```

These initial steps are shown in **Figure 6**.

Here are the final actions of the Logic App:

Parse JSON will evaluate the contents of the blob and provide a set of variables I can reference later. The expression for the Content property is:

```
json(body('Get_blob_content_using_path'))
```

I'll also need to provide a sample payload that looks like:

```
[{
  "id": "100",
  "eventTime": "2017-08-21T06:42:20.0000000+00:00",
  "eventType": "type",
  "dataVersion": "",
  "metadataVersion": "1",
  "topic": "enpoint",
  "subject": "testsubject",
  "deadLetterReason": "reason",
  "deliveryAttempts": 1,
  "lastDeliveryOutcome": "BadRequest",
  "lastHttpStatusCode": 400,
  "data": { "something": "data" }
}]
```

Send an email is the last action, which formats the body and

subject of the email with the artifacts of the events (dead letter reason and subject). Because the payload actually contains an array, Logic Apps recognizes that and wraps the action in a for each action.

As a side note, each step within a for each loop is done in parallel instead of sequentially in Logic Apps; this is the default behavior that can be changed to run sequentially, if desired. The end result is displayed in **Figure 7**.

Testing this end-to-end will result in an email being sent for each dead letter event.

Wrapping Up

With its original design and deep integration into a growing list of Azure services, Event Grid is only beginning to reveal innovative ways to build event-driven solutions. In this article, I demonstrated how to leverage the new retry policies and dead lettering functionality in Azure Event Grid. These highly anticipated features are powerful options for making solutions built on Event Grid more resilient and scalable. The code in this article can be found at bit.ly/2LGKjhN. ■

DAVID BARKOL is an Azure specialist at Microsoft on the Global Black Belt Team. You can contact him on Twitter: @dbarkol or through email at dabarkol@microsoft.com. He blogs regularly about Event Grid at madeofstrings.com.

THANKS to the following Microsoft technical expert for reviewing this article: Bahram Banisadr

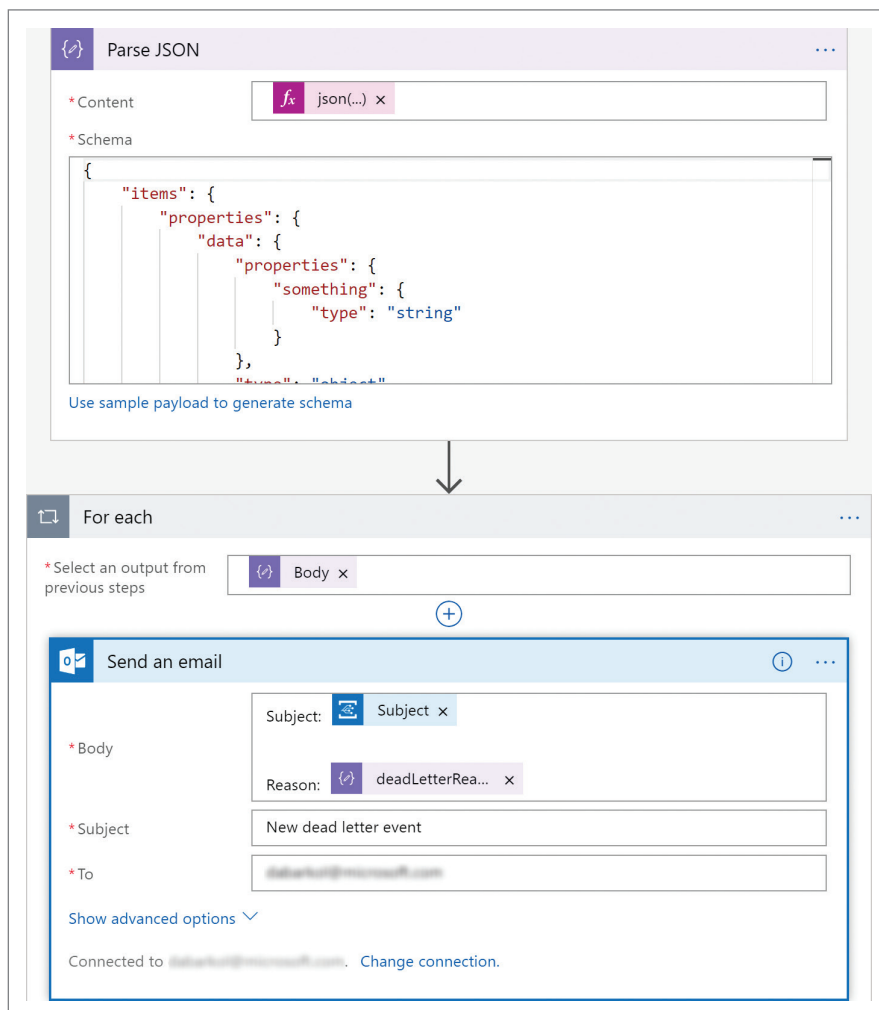
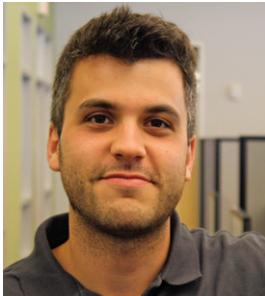


Figure 7 Parsing JSON and Sending Notifications



LEADTOOLS Imaging SDKs Modeling Excellence and Dependability

A Q&A with Hadi Chami, Developer Support Manager

What is new in LEADTOOLS Version 20?

Our biggest goal for Version 20 was to make LEADTOOLS available on every major platform. To accomplish this, we extended our technology with new libraries leveraging .NET Standard targeting .NET Framework, .NET Core, and Xamarin app models.

With these new platforms also came new delivery mechanisms. A majority of our .NET libraries are now available via NuGet packages. NuGets were heavily requested by our customer base and we made sure to provide thoughtfully bundled feature groups developers are already using. Putting LEADTOOLS on NuGet.org helps customers get coding faster and stay up to date with current builds.

LEADTOOLS is a pretty comprehensive toolkit.

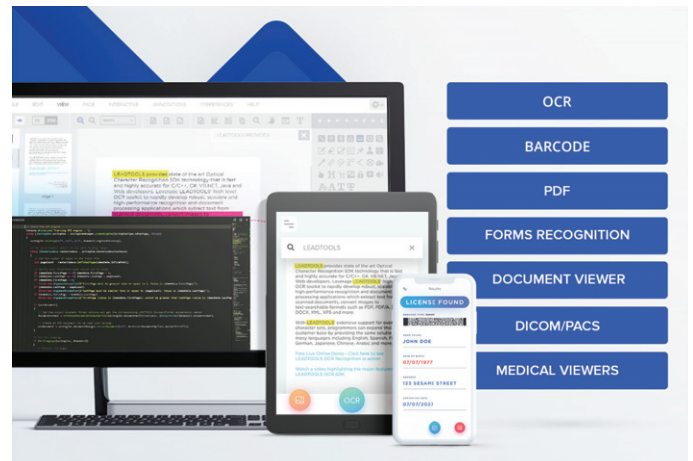
What could you say LEADTOOLS is best known for?

That's a tough question! Like you said, LEADTOOLS is very comprehensive with SDKs for Document, Medical, Multimedia, and Raster Imaging. In recent years, OCR and Document Viewing have been our top performing technologies. We have our own, in-house developed OCR engine we are constantly improving, as opposed to putting a stagnant wrapper around a 3rd party engine. The LEAD engine is very fast and accurate, and our head OCR developer just finished up his PhD. thesis on character recognition.

Our HTML5/JavaScript Document Viewer is also a headlining component for us. The modern workforce relies heavily on digital documents, but it's not just TIFF and PDF anymore. Word, Excel, RTF, Text, PNG, and JPEG files are commonly in the mix whenever you look at someone's cloud storage or ECM. Our Document Viewer can treat all formats uniformly, allowing you to search text, convert, save/edit annotations, etc. Best of all, as a web application that runs on Windows or Linux servers, it's incredibly secure and adaptable, providing scalable zero-footprint document viewing to any desktop, tablet, or smart phone with a web browser.

What sets LEADTOOLS apart from other commercial and open-source SDKs?

Finding the right SDK is tough enough, but many applications need multiple libraries. LEADTOOLS is broad enough for you to standard-



ize on LEADTOOLS for all imaging requirements and eliminate multiple-vendor headaches.

Stack Overflow is great, but sometimes you need a personal touch. Our Support can prove to be a huge advantage over many open-source options where you are largely on your own. LEADTOOLS isn't a side-project developed in our spare time, it's our core product. We are a twenty-eight-year-old company, and are committed to building, maintaining, and supporting our technology.

How is LEADTOOLS adapting to the new ways developers are writing applications?

In addition to offering NuGet packages, LEAD created LEADTOOLS Cloud Services, an alternative development framework. This high-powered, scalable Web API gives developers a hassle-free interface for integrating advanced recognition and document conversion into any application. LEADTOOLS Cloud Services provides all the speed, accuracy, and reliability you have come to expect from LEADTOOLS but is neatly packaged into a simple, pay-as-you-go Web API hosted on Microsoft Azure. With minimal requirements, developers can use their API key to make JSON requests via several programming languages, including PHP, Python, and Perl. Additionally, a wider array of tools and IDEs can be used including Visual Studio Code, Sublime, and Notepad++.

Visual Studio® LIVE!

EXPERT SOLUTIONS FOR .NET DEVELOPERS

September 17-20, 2018
Renaissance

Chicago

Visual Studio LIVE
25
YEARS OF CODING INNOVATION
1993 - 2018

Look Back to Code Forward

Visual Studio Live! (VSLive!™) is thrilled to be returning to Chicago where developers, software architects, engineers and designers will “look back to code forward” during four days of unbiased and cutting-edge education on the Microsoft Platform.

Tackle training on the hottest topics (like .NET Core, Angular, VS2017), debate with industry and Microsoft insiders (people like Rockford Lhotka, Deborah Kurata and Brock Allen) and network with your peers—plus, help us celebrate 25 years of coding innovation as we take a fun look back at technology and training since 1993. Come experience the education, knowledge-share and networking at #VSLive25.

DEVELOPMENT TOPICS INCLUDE:



**DevOps in the
Spotlight**



**Cloud, Containers
and Microservices**



**AI, Data and
Machine Learning**



**Developing New
Experiences**



**Delivery and
Deployment**



**.NET Core
and More**



**Full Stack
Web Development**



vslive.com/chicago

**Best Savings End
September 17**

Use Promo Code MSDN

GOLD SPONSOR



EXHIBITOR



SUPPORTED BY



PRODUCED BY



DevOps in the Spotlight		Cloud, Containers and Microservices		AI, Data and Machine Learning		Developing New Experiences		Delivery and Deployment		.NET and More		Full Stack Web Development	
START TIME	END TIME	Pre-Conference Workshops: Monday, September 17, 2018 <small>(Separate entry fee required)</small>											
8:00 AM	9:00 AM	Pre-Conference Workshop Registration - Coffee and Morning Pastries											
9:00 AM	6:00 PM	M01 Workshop: Build a Modern ASP.NET App in the Cloud with a full CI/CD Pipeline in VSTS - <i>Brian Randell</i>				M02 Workshop: SQL Server for Developers - <i>Andrew Brust and Leonard Lobel</i>				M03 Workshop: Distributed Cross-Platform Application Architecture - <i>Rockford Lhotka and Jason Bock</i>			
6:45 PM	9:00 PM	Dine-A-Round											
START TIME	END TIME	Day 1: Tuesday, September 18, 2018											
7:00 AM	8:00 AM	Registration - Coffee and Morning Pastries											
8:00 AM	9:15 AM	T01 The State of XAML: Recent Changes for UWP, WPF, Xamarin - <i>Billy Hollis</i>			T02 An Introduction to TypeScript - <i>Jason Bock</i>			T03 SQL Server Security Features for Developers - <i>Leonard Lobel</i>			T04 Get Started with Git - <i>Robert Green</i>		
9:30 AM	10:45 AM	T05 Building Your First Mobile App with Xamarin Forms - <i>Robert Green</i>			T06 Essential Web Development with ASP.NET Core - <i>Mark Michaelis</i>			T07 Exploring T-SQL Enhancements: Windowing and More - <i>Leonard Lobel</i>			T08 DevOps for the SQL Server Database - <i>Brian Randell</i>		
11:00 AM	12:00 PM	KEYNOTE: The Present and Not Too Distant Future of Visual Studio - <i>Amanda Silver, Partner Director of Program Management, Microsoft</i>											
12:00 PM	1:30 PM	Lunch											
1:30 PM	2:45 PM	T09 Cross-Platform App Dev with C# and CSLA .NET - <i>Rockford Lhotka</i>			T10 Assembling the Web—A Tour of WebAssembly - <i>Jason Bock</i>			T11 Glue for the Internet: Introducing Azure Event Grid - <i>Jeremy Likness</i>			T12 Azure DevOps with VSTS, Docker, and K8 - <i>Brian Randell</i>		
3:00 PM	4:15 PM	T13 A Dozen Ways to Mess Up Your Transition From Windows Forms to XAML - <i>Billy Hollis</i>			T14 Entity Framework Core 2 For Mere Mortals - <i>Philip Japikse</i>			T15 Code First in the Cloud: Serverless .NET with Azure - <i>Jeremy Likness</i>			T16 Essential C# 8.0 - <i>Mark Michaelis</i>		
4:15 PM	5:30 PM	Welcome Reception											
START TIME	END TIME	Day 2: Wednesday, September 19, 2018											
7:30 AM	8:00 AM	Registration - Coffee and Morning Pastries											
8:00 AM	9:15 AM	W01 Electron: Desktop Development For Web Developers - <i>Chris Woodruff</i>			W02 JavaScript for the C# (and Java) Developer - <i>Philip Japikse</i>			W03 Quantum Computing and the Future of Software Development - <i>Jerry Nixon</i>			W04 Building a Stronger Team, One Strength at a Time - <i>Angela Dugan</i>		
9:30 AM	10:45 AM	W05 Enhancing UWP Experiences with Fluent Design - <i>Tony Champion</i>			W06 Architecting and Developing Microservices Apps - <i>Eric D. Boyd</i>			W07 Sharing C# Code Across Platforms - <i>Rockford Lhotka</i>			W08 How do You Measure up? Collect the Right Metrics for the Right Reasons - <i>Angela Dugan</i>		
11:00 AM	12:00 PM	GENERAL SESSION: Microsoft's Journey to DevOps - <i>Jay Schmelzer, Director of Program Managemnet, Visual Studio Team Microsoft</i>											
12:00 PM	1:00 PM	Birds-of-a-Feather Lunch											
1:00 PM	1:30 PM	Dessert Break - Visit Exhibitors - Exhibitor Raffle @ 1:15pm (Must be present to win)											
1:30 PM	2:45 PM	W09 Learning The Language Of HTTP For A Better Data Experience In Your Mobile Apps - <i>Chris Woodruff</i>			W10 Angular 101 - <i>Deborah Kurata</i>			W11 Power BI: What Have You Done for Me Lately? - <i>Andrew Brust</i>			W12 Fault Driven Development - <i>Josh Garverick</i>		
3:00 PM	4:15 PM	W13 Building Cross Device Experiences with Project Rome - <i>Tony Champion</i>			W14 N Things You Didn't Know About the Router - <i>Deborah Kurata</i>			W15 Analytics and AI with Azure Databricks - <i>Andrew Brust</i>			W16 Core Azure Solutions: Automation - <i>Josh Garverick</i>		
4:30 PM	5:45 PM	W17 Use UWP to Modernize Your Existing WinForms and WPF Applications - <i>Walt Ritscher</i>			W18 Tools for Modern Web Development - <i>Ben Hoelting</i>			W19 Create Intelligent Bots with Cognitive Services and Azure Search - <i>Eric D. Boyd</i>			W20 Real World Scrum with Team Foundation Server & Visual Studio Team Services - <i>Benjamin Day</i>		
6:30 PM	9:00 PM	VSLive's Windy City Sunset Cruise											
START TIME	END TIME	Day 3: Thursday, September 20, 2018											
7:30 AM	8:00 AM	Registration - Coffee and Morning Pastries											
8:00 AM	9:15 AM	TH01 PowerShell for Developers - <i>Brian Randell</i>			TH02 Docker for ASP.NET Core Developers - <i>Michele Leroux Bustamante</i>			TH03 MVVM and ASP.NET Core Razor Pages - <i>Ben Hoelting</i>			TH04 Unit Testing & Test-Driven Development (TDD) for Mere Mortals - <i>Benjamin Day</i>		
9:30 AM	10:45 AM	TH05 From Waterfall to Agile. Microsoft's Not-So-Easy Evolution into the World of DevOps - <i>Abel Wang</i>			TH06 Developing Microservices Solutions on Azure - <i>Michele Leroux Bustamante</i>			TH07 Eliminate Code Using Data Binding in WPF - <i>Paul Sheriff</i>			TH08 C# 7, Roslyn and You - <i>Jim Wooley</i>		
11:00 AM	12:15 PM	TH09 Writing Testable Code and Resolving Dependencies—DI Kills Two Birds with One Stone - <i>Miguel Castro</i>			TH10 Effective Data Visualization - <i>David Giard</i>			TH11 Store Data Locally for Offline Web Applications - <i>Paul Sheriff</i>			TH12 Improving Code Quality with Static Analyzers - <i>Jim Wooley</i>		
12:15 PM	1:15 PM	Lunch											
1:15 PM	2:30 PM	TH13 Exposing an Extensibility API for Your Applications and Services - <i>Miguel Castro</i>			TH14 Adding Image and Voice Intelligence to Your Apps with Microsoft Cognitive Services - <i>David Giard</i>			TH15 Modern Security Architecture for ASP.NET Core - <i>Brock Allen</i>			TH16 SQL Server 2017—Intelligence Built-in - <i>Scott Klein</i>		
2:45 PM	4:00 PM	TH17 Advanced DevOps—Deep Dive into Feature Flags - <i>Abel Wang</i>			TH18 Programming with Microsoft Flow - <i>Walt Ritscher</i>			TH19 Implementing Authorization in Web Applications and APIs - <i>Brock Allen</i>			TH20 Databases and Data Lakes—Bridging the Gap - <i>Scott Klein</i>		

Speakers and sessions subject to change

CONNECT WITH US



twitter.com/vslive – @VSLive



facebook.com – Search "VSLive"



linkedin.com – Join the "Visual Studio Live" group!

vslive.com/chicago

Architect Blockchain Smart Contracts as Microservices

Stefano Tempesta

Microservices and blockchain smart contracts have a lot in common. They're both expected to run in isolation (on-chain) and communicate with the outside (off-chain) via a message-based channel. They should both be small in size, developed to run autonomously and independently, and perform better when they're deployed on a decentralized network.

This article presents design principles, artifacts and code samples for building blockchain applications using a microservice architecture style and deploying them on the Microsoft Azure Blockchain platform.

Microservices perfectly embody the spirit of the Unix philosophy: Do one thing and do it well (tcrn.ch/2vnq5Pb). A microservice is an independent, deployable component of bounded scope that supports interoperability through message-based communication. Given this premise, microservice architecture is a style of

engineering that helps build highly automated, evolvable software systems made up of single-capability microservices.

What do blockchain applications have in common with microservices, and what design principles can be applied from microservice architectures to the decentralized world? The table in **Figure 1** compares microservices and smart contracts against specific design attributes.

Designing blockchain applications as microservices can bring the following benefits to your solution:

- Allow many software engineering initiatives to run in parallel.
- Lessen dependencies between software development and test teams.
- Support multiple technologies, languages and frameworks.
- Promote ease of innovation through disposable code.

Microservices typically speak to the outside world via an application programming interface (API) that shares a common language, say JSON or SOAP, with the client—providing a lingua franca of messaging-enabled systems across different technologies (.NET, Java, Node.js and so on) and platforms (Windows, Linux). This is true also for the blockchain API exposed by Azure Blockchain Workbench, as you'll see later in this article.

From Microservices to Decentralized Applications

If you're familiar with the DevOps sentiment of treating your servers like cattle and not pets (bit.ly/2vrdM4p), you might apply the same approach to your source code. Easily disposable code can reduce technical debt, promote modernization of engineering processes and decrease operational costs by optimizing infrastructure (for example, in containers or entirely serverless configurations).

This article discusses:

- Microservice architecture design
- Decentralized domain-driven design
- Event sourcing and CQRS
- Distributed transaction
- Azure Blockchain Workbench API
- Asynchronous messaging

Technologies discussed:

Microservices, Blockchain, Ethereum, Microsoft Azure Blockchain Workbench

Figure 1 Microservices and Blockchain Design Principles

Design Principle	Microservice	Smart Contract
Single responsibility	Typically provides a CRUD interface on a single entity.	Defines roles, state and the relevant logic for a validation workflow on a single object, using the CRAB approach (more later in this article).
Context bounded	No dependency on other services, owns its data model for persistent storage.	Does not have dependency on other smart contracts and leverages the on-chain data model (that is, the blockchain itself) as the preferred data model.
Messaging enabled	Can leverage an API gateway for inter-service communication, and a service bus for intra-service communication.	Can leverage “oracles” or “cryptlets” for off-chain data access, or tools like Azure Blockchain Workbench that expose a REST API.
Autonomously developed	Multiple programming languages and frameworks.	Multiple blockchain platforms available, although no cross-platform communication currently exists.
Independently deployable	With proper design (event sourcing, CQRS) can reduce or remove dependency completely.	Similar design patterns apply (described in this article).
Distributed and decentralized	Distributed architecture as opposed to a centralized “monolith.”	Built-in distributed and decentralized digital ledger by design.

Designing blockchain applications with microservice architecture principles can also yield business benefits. Improved efficiency in the software system reduces infrastructure costs and the risk of capacity-related service outages. These aspects are of particular value to private blockchains, where cost effectiveness and service continuity are key requirements for businesses.

Microservice architecture principals can support use of replaceable components, reducing technical debt that can lead to aging, unreliable environments. Solidity, the programming language for smart contracts in Ethereum, has a mechanism for specifying the exact runtime version on which each contract is executed. Over time, a smart contract’s version number can be used to identify obsolete blockchain-stored code that may be a candidate for replacement. Just keep in mind that in a blockchain, smart contracts that have already been processed (that is, are part of a “mined” block) cannot be deleted—a new version of a contract must be published for future transactions.

Another benefit is better runtime scalability, which allows a software system to grow or shrink with demand. Smart contracts implemented as microservices enable permissioned blockchains in a private business or consortium environment to distribute workloads for transaction and mining nodes in a more flexible way.

At its most basic level, microservice architecture is about breaking up an application or system into smaller parts and gaining benefit from the distributed setup. In the same vein, smart contracts that run on a blockchain benefit from the distributed nature of the peer-to-peer network. With a microservice architecture-oriented design, smart contracts can deliver improved efficiency, scalability and manageability—all attributes that are essential for proper implementation of blockchain solutions in the enterprise.

Decentralized Domain-Driven Design

Write an application for a decentralized blockchain digital ledger and you’re working with infrastructure and software requirements typical of a distributed system, such as storage isolation, asynchronous messaging and distributed transactions. Blockchain applications also require authentication of users and devices, and authorization to execute specific actions within a smart contract. Expanding on the popular Domain-Driven Design (DDD) approach, I refer to the practice of considering entities, processes and properties of a blockchain system as “Decentralized Domain-Driven Design,” or DDDD.

Let’s start defining the domain, or context, of the execution of smart contracts within a digital ledger. Smart contracts represent the business logic of the blockchain application as workflows, with each stage of a workflow identified by a message sender (a user or device that executes a function of the contract) and a state (the parameters of the contract, represented as a message sent to a function, and its internal status). Other parties (again, users or devices) may be affected by the execution of a function of a smart contract.

Each contract contains roles, state and functions to implement actors, stages and actions of a business process.

In this context, we refer to all parties involved as application roles. The very first application role that creates the contract is called the initiator. On change of internal state of a contract, an event may be raised to signal this change to other parts of the smart contract, or to external applications. This is a typical pattern, for example, for populating off-chain data, by using a service bus for processing events raised by a smart contract and publishing messages to the relevant listeners.

Figure 2 identifies the entities involved in a workflow within a smart contract.

Ethereum uses Solidity as its programming language for writing self-enforcing business logic for smart contracts. Smart

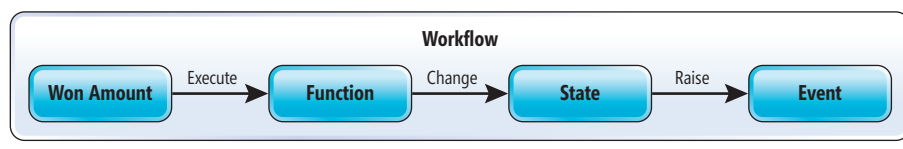


Figure 2 Workflow Entities in a Smart Contract

Figure 3 Declaration of Roles, State and Properties in the Betting Smart Contract

```
pragma solidity ^0.4.20;

contract Betting
{
    // Roles
    address public Gambler;
    address public Bookmaker;

    // State
    enum BetType { Placed, Won, Lost }
    BetType public State;

    // Properties
    uint public BetAmount;
}
```

contracts in Solidity are similar to classes in object-oriented languages. Each contract contains roles, state and functions to implement actors, stages and actions of a business process.

The code snippet in **Figure 3** shows the different types of variable declaration in Solidity for roles, state and properties that may be used in a smart contract for a betting application. Roles (the gambler and the bookmaker) are defined as address, which is the unique identifier of a user or contract in Ethereum. State is an enumeration of labels that identifies the current state of a bet placed via the smart contract. Functions, as you'll see later, define a change of state. The bet amount is expressed as an unsigned number (currently, Solidity doesn't support decimal values).

Please note the indication of the version of Solidity that I'm targeting for this smart contract. It's good practice to indicate this pragma instruction to avoid incompatibilities with future versions of the Solidity programming language and compiler. This also helps identify old code in a smart contract, which may need to be replaced with a new version to support updated code. The process of removing an existing smart contract from a blockchain is called "self-destruct."

A smart contract should have a single responsibility and contain as little business logic as possible—optimally only the validation logic needed to deem a contract valid or not. In my betting application, a smart contract may expose functions for placing a bet by the gambler, and acknowledging win or loss by the bookmaker. Currency may be exchanged between the two roles, as part of the bet workflow. The usual pattern for monetary transactions that require validation by a contract sees the amount transfer happening in two phases, as indicated in **Figure 4**. The gambler (initiator of the contract) places a bet for a certain amount, which is then stored in the smart contract. If the bet is won, the won amount, indicated by the bookmaker, is transferred to the gambler. Otherwise, the bookmaker cashes the bet amount.

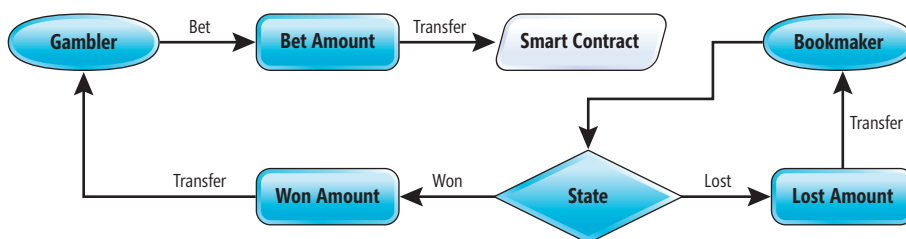


Figure 4 Betting Workflow

As shown in **Figure 5**, the implementation of this smart contract in Solidity requires that several functions be defined for each action of the workflow, as follows:

- The constructor stores the message sender as the gambler; this is the role that initiates the smart contract.
- The Bet function accepts an amount as input, performs some validation (this function can be called only by the gambler and the amount should be more than zero), and then transfers the bet amount to the contract. To allow for on-chain currency transfers, it's necessary to flag a function as payable.
- The Won function, after validating that the invoker isn't the gambler, transfers the won amount to the gambler and closes the bet as "Won."
- The Lost function, which again can only be invoked by the bookmaker, transfers the amount initially bet, and now lost by the gambler, to the bookmaker and closes the bet as "Lost."
- By closing a bet, the gambler is removed (its address is set to 0x0) and the amount set to zero, ready for another bet.

A smart contract should have a single responsibility and contain as little business logic as possible—optimally only the validation logic needed to deem a contract valid or not.

Though simple in implementation, this scenario identifies a typical pattern for managing monetary spending in a blockchain application. Other scenarios may need to incorporate attestable files, such as documents, spreadsheets, certificates and pictures. For multiple reasons, mainly concerning storage limitation, it's inappropriate to put files on a blockchain. A common approach is to perform a cryptographic hash (for example, SHA-256) against a file and share that hash on a distributed ledger. The external system would instead persist the file into a storage mechanism, such as Azure Storage or IPFS (ipfs.io).

Performing the hash again with the same hashing algorithm at any future time will return the same result, unless the persisted file was modified—even if just one pixel is modified in an image. This process grants proof of existence that an information object like

an email, file, document, phone call or video existed at a certain point in time. It also grants proof of authenticity—you know a digital asset hasn't changed because the digital ledger stores immutable and independent, verifiable records of all transactions. For more on the value of blockchain in enterprise content management, read the post I published at bit.ly/20C2Ycp.

Figure 5 Functions in the Betting Smart Contract

```

constructor() public {
    Gambler = msg.sender;
}

function Bet(uint amount) public payable {
    require(msg.sender == Gambler, "Only a gambler can place a bet.");
    require(amount > 0, "Amount should be greater than zero.");

    BetAmount = amount;
    address(this).transfer(amount);

    State = BetType.Placed;
}

function Won(uint amount) public payable {
    require(msg.sender != Gambler, "Only the bookmaker can mark a bet as won.");
    require(amount > 0, "Amount should be greater than zero.");

    Gambler.transfer(amount);
    Close(BetType.Won);
}

function Lost() public payable {
    require(msg.sender != Gambler, "Only the bookmaker can mark a bet as won.");

    Bookmaker = msg.sender;
    Bookmaker.transfer(BetAmount);
    Close(BetType.Lost);
}

function Close(BetType state) internal {
    Gambler = 0x0;
    BetAmount = 0;
    State = state;
}

```

Event Sourcing and CQRS

As discussed in the previous section, I recommend building smart contracts with responsibility for one capability only. While capability-oriented design is a crucial technique for isolation of smart contracts, it's not sufficient to ensure independent deployability. Smart contracts may operate on a common data model within the domain of the system, despite being isolated in execution. For example, in an application there may be a smart contract for managing bets and another for managing sporting events on which to bet. The smart contract for betting may reference a sporting event, creating a dependency between the two smart contracts (betting

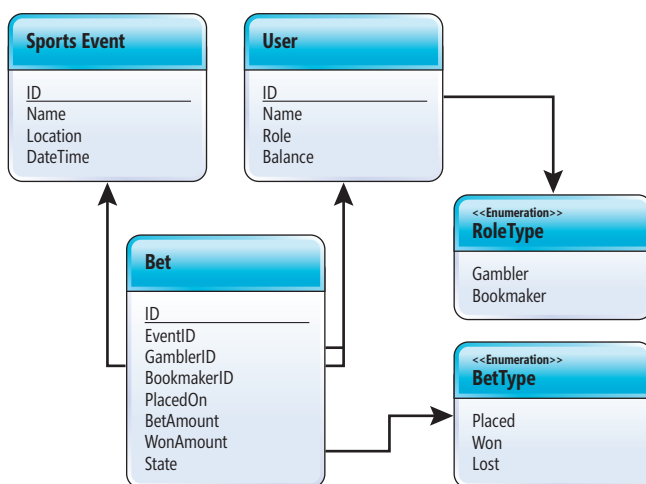


Figure 6 Structural Data Model

cannot happen if an event doesn't exist). Is there a way to model data that can help avoid data sharing among smart contracts?

Whatever format and storage (SQL, NoSQL, JSON) we use, we generally model databases according to objects and Create, Read, Update, Delete (CRUD) operations. Instead of storing structures that model the state of the domain, we can store events that lead to the current state of our world. This modeling approach is called event sourcing (bit.ly/2068nrt).

Event sourcing is all about storing facts. A fact is a representative value of an event occurrence. Just as in life, we can't go back in time and change the past—we can only do something in the present to compensate for earlier actions. Data is immutable; so we always issue a new command or event to compensate for, rather than update, the state of an entity. This approach operates under the acronym of CRAB—Create, Retrieve, Append and Burn (bit.ly/2MbpU0b), which is exactly what a blockchain allows to execute: no data updates or deletions, it only appends to the chain. Deleting something from a blockchain conflicts with its immutability, but you can stop asset transfer by “burning” the recipient address.

An immediate concern of this approach is performance. If any state value is a function of events, you may assume that every access to the value would require recalculation of the current state from the source events. Obviously, that would be extremely slow. In event sourcing, you can avoid such expensive operations by using a so-called rolling snapshot: a projection of the entity state at a given point in time. For instance, banks pre-calculate your bank account balance on the last day of every month, so you don't need to sum all debits and credits operations since the day you opened your bank account to obtain your current balance.

Figure 6 shows the structural data model for the betting application. This is sometimes called the snowflake model, because each entity (a database table) is different from any other.

The structural data model saves only the current state of the system, while the event-sourcing approach saves individual facts. State, in event sourcing, is a function of all the pertinent facts that occurred. Not only does this give full auditability, but it also allows us to build state projections toward any time in the past.

To push this further in terms of isolation of responsibilities, Command Query Responsibility Segregation (CQRS) complements event sourcing as a design pattern for data storage. CQRS encourages effective single responsibility and deployability of microservices, and by extension, smart contracts. It states that you can—and should—separate data-update from data-query capabilities into separate models.

When using CQRS, the need to access data across multiple contexts can be eliminated. A smart contract can own and encapsulate



Figure 7 Event-Sourcing Data Model

TEXTCONTROL

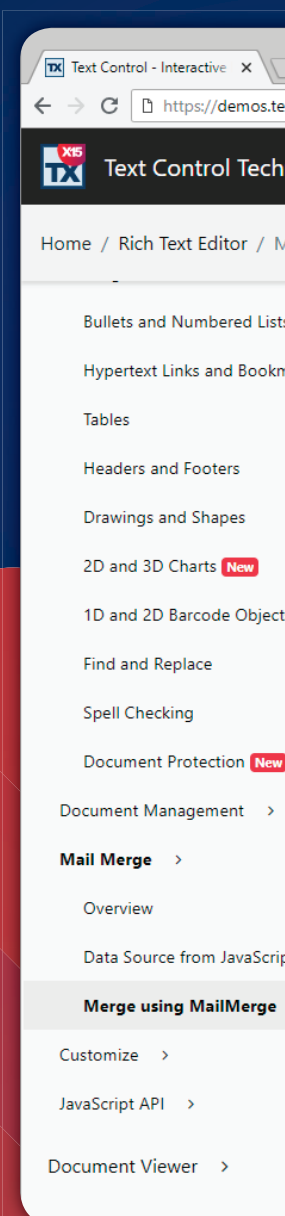
Integrate Documents and Reporting in Any Platform

The Text Control Reporting Framework combines powerful reporting features with an easy-to-use, MS Word compatible, word processor.

See our technology live:

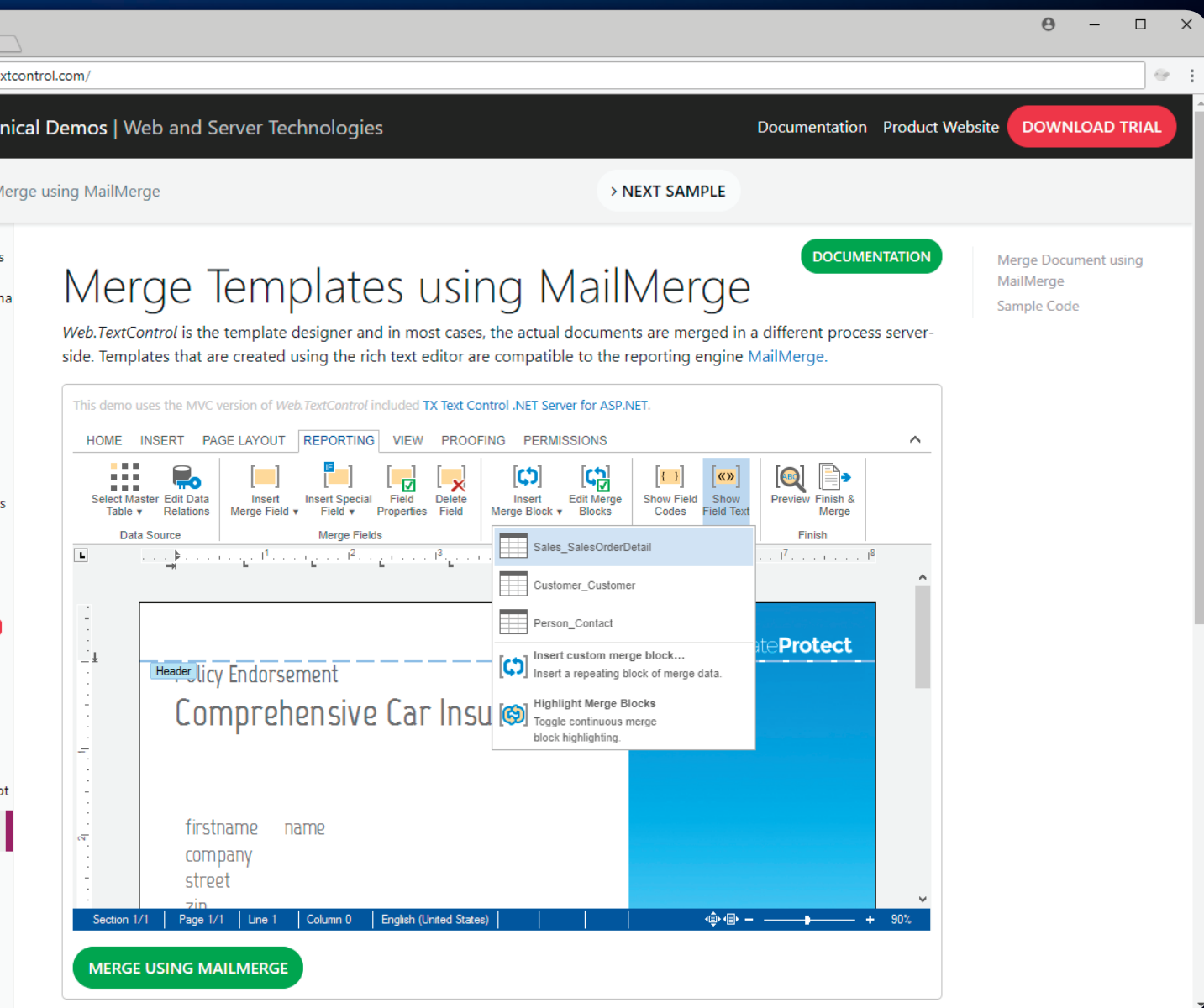
demos.textcontrol.com

**WE ARE CHANGING
THE WAY YOU LOOK AT
REPORTING**



TX Text Control X15 Technical Demos

Evaluate our technology and test the most sophisticated, cross-browser, and true WYSIWYG, rich text editor. Merge MS Word compatible templates with JSON data and create pixel-perfect Adobe PDF documents on-the-fly. See what's possible today!



any update to the model's state and raise events on change of this state. By subscribing to notifications of these events, a separate smart contract can build a completely independent and query-optimized model that doesn't need to be shared with any other contract or external service. You can learn more about CQRS from Martin Fowler's post at bit.ly/2Awoz33.

Figure 7 describes the data model that I designed for the betting application using event sourcing. This simple model employs a similar structure regardless of the event handled. It's not necessary to know the current state of the bet to read the sequence of events. The event's data structure depends on the event itself. Although a sequence of states exists as defined in the workflow, it's irrelevant from a data-model perspective. Think bigger: In a supply-chain scenario, multiple workflows and events exist, with different entities and attributes. Your structural data model may grow complex, whereas the event-based model, bar a few different attributes for each event, remains constant.

Distributed Transactions

A shared data model isn't the only use case that can introduce tight coupling between smart contracts. Another important threat is workflows. A lot of real-life processes cannot be represented with a single, atomic operation. With such workflows, the result only makes sense if all the steps can be executed. If any step in the sequence fails, the resulting state of the relevant system becomes invalid. In the RDBMS world, such processes are called "transactions." Database transactions are typically local, contained within the confines of a single database, and rely on locks on tables before updates. If a step fails, you can roll back the steps already attempted before a final commit.

For distributed workflows and stateless microservices, a traditional transaction implementation with data locks and Atomicity, Consistency, Isolation, Durability (ACID) compliance (en.wikipedia.org/wiki/ACID) is impractical. Sagas (bit.ly/2AzdKNR) are long-lived distributed transactions that allow running workflows in loosely coupled environments, without making any assumption of the reliability of each component of the complex system.

In sagas, every step in the workflow executes its portion of the work, registers a call back to a compensating transaction in a message called "routing slip" and passes the updated message down the activity chain. If any step downstream fails, that step looks at the routing slip and invokes the most recent step's compensating transaction, passing back the routing slip. The previous step does the same thing, calling its predecessor's compensating transaction and so on until all already executed transactions are compensated. This pattern leads to eventual consistency of data in a distributed transaction (bit.ly/2v8T360). Due to its highly fault-tolerant, distributed nature, sagas are very well-suited to a microservice architecture, as well as to blockchain smart contracts.

You can implement a sort of routing slip via fallback functions in Solidity. A fallback function is an unnamed function defined with no input argument and no return value. It's executed on a call to the contract if none of the other functions match the given function identifier or whenever the contract receives Ether (in the case of Ethereum). Additionally, in order to receive Ether, the fallback function must

be marked payable. If no such function exists, the contract cannot receive Ether through regular (address-to-address) transactions.

It's worth mentioning that a contract without a payable fallback function can receive Ether as a recipient of a coinbase transaction, such as a miner block reward. A contract cannot react to such Ether transfers and, thus, also cannot reject them. This is a design choice of Ethereum, and Solidity cannot work around it. A contract can have exactly one unnamed function, as shown here:

```
// Fallback function
function() public payable {
    emit AmountTransferred(msg.sender);
}

event AmountTransferred(address sender);
```

In Ethereum, fallback functions are necessary for a smart contract to allow account-to-account direct transfers. This is because the transferring account may need to make transfers to both Externally Owned Accounts (EOAs) and to other smart contracts. As EOAs can only accept direct transfers, the only way for an account to transfer value to another account is for the executing contract to implement a fallback function. This means that any contract that wants to accept such transfers must be prepared for direct transfers by having a fallback function. Without that function, the transfer would fail, and it would be impossible for the contract to accept Ether from the other contract.

Asynchronous messaging
plays a key role in keeping
things loosely coupled
in a microservice architecture.

A best practice is to not have any logic in the fallback function. It's possible to put code in the body of this function, but it's best to avoid anything beyond very short, simple logging. The reason is important and unique to smart contracts: You don't want this function to fail because it runs out of gas. As a rule of thumb, you'll have just enough gas to raise an event, but not enough to write data to storage.

Asynchronous Messaging

Asynchronous messaging plays a key role in keeping things loosely coupled in a microservice architecture. For example, we can use a message broker to deliver event notifications in an asynchronous manner, preventing point-to-point connections that create a dependency on each endpoint availability and message format. By the same token, smart contracts can benefit from messaging-enabled integration for inbound (from outside to inside the blockchain) and outbound (from the blockchain toward external applications) communication.

In addition to providing a REST API, Azure Blockchain Workbench provides messaging-based integration based on ledger-centric events. Events are published to an Azure Event Grid, and consumers can ingest data or take action based on these events. For those clients that require reliable messaging, Azure Blockchain Workbench



TECH EVENTS WITH PERSPECTIVE



Artificial Intelligence LIVE!

AI FOR DEVELOPERS AND DATA SCIENTISTS

ROYAL PACIFIC RESORT AT
UNIVERSAL ORLANDO
December 2-7, 2018

Full Agenda
Available Now!
ATTENDAILIVE.COM

A New AI Event for Developers and Data Scientists

Artificial Intelligence Live! is an innovative, new conference for current and aspiring developers, data scientists, and data engineers covering artificial intelligence (AI), machine learning, data science, Big Data analytics, IoT & streaming analytics, bots, and more. You can expect real-world training on the languages, libraries, APIs, tools and cloud services you need to implement real AI and machine learning solutions, today and into the future.

TRACK TOPICS INCLUDE:

- Artificial Intelligence (AI)
- Machine Learning
- Data Science
- Big Data analytics
- IoT & Streaming Analytics
- Bots
- & More....

REGISTER
NOW

SAVE \$400 WITH SUPER EARLY BIRD PRICING! REGISTER BY OCTOBER 5

Use Promo Code MSDN

A Part of Live! 360: The Ultimate Education Destination

6 CONFERENCES, 1 GREAT PRICE

Visual Studio LIVE! | SQL Server LIVE! | **TECHMENTOR** | Artificial Intelligence LIVE! | Office & SharePoint LIVE! | Modern Apps LIVE!



ATTENDAILIVE.COM

EVENT PARTNERS



SILVER SPONSOR



PREMIER & ALLIANCE MEDIA PARTNERS



SUPPORTED BY



PRODUCED BY



delivers messages to an Azure Service Bus endpoint, as well. You can raise events, for example, to notify users and systems of transactions or changes of state in a smart contract. Event notifications can be consumed directly in code or used with tools such as Logic Apps (bit.ly/2n4EgoP) to trigger flow of data to downstream systems.

Smart contracts often represent a business workflow that integrates with external systems and devices. As a result, transactions must be able to initiate on a distributed ledger that includes data from an external system or device. You may also want to have external systems react to events originating from smart contracts on a distributed ledger. The REST API and messaging integration provide the ability to both send transactions from external systems to smart contracts included in an Azure Blockchain Workbench application, and send event notifications to external systems based on changes that take place within an application. Let's explore now the patterns identified for each of these types of integrations in your end-to-end solutions:

- One-way event delivery from a smart contract to an event consumer.
- One-way event delivery of a message from an external system to a smart contract.

Smart Contract to an Event Consumer

In the first scenario, an event occurs within a smart contract; for example, a state change or the execution of a specific type of transaction. This event is broadcast via an Event Grid to downstream consumers, and those consumers then take appropriate actions. An example of this scenario is that when a transaction occurs, a consumer would be alerted and could take action, such as recording the information in a database. This is the same pattern that Azure Blockchain Workbench follows to populate its off-chain SQL database. Another would be if a smart contract transitions to a particular state; for example, when a contract goes into an "Out of Compliance" state. When this state change happens, it could trigger an alert to be sent to an administrator. This occurs using the process depicted in **Figure 8**, where:

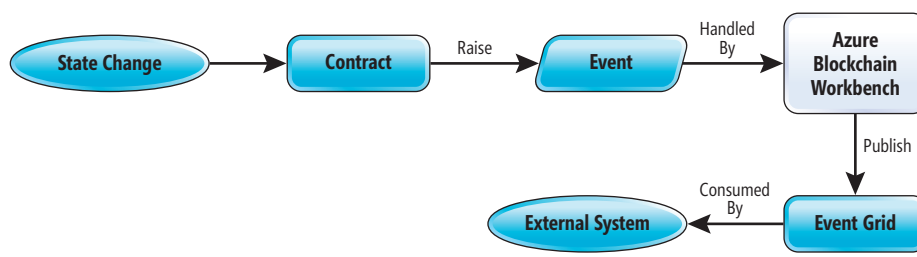


Figure 8 Propagation of an Event Raised in a Smart Contract to an External System

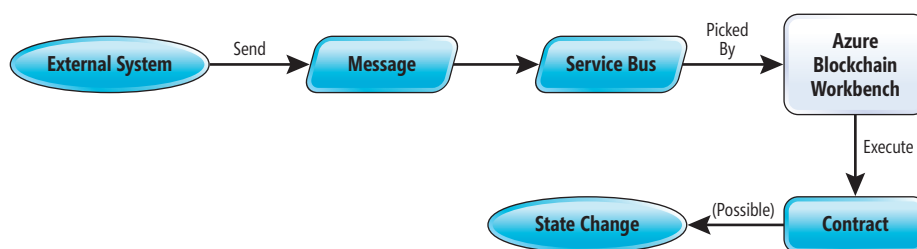


Figure 9 Propagation of an Event Raised by an External System to a Smart Contract

1. The smart contract transitions to a new state and sends an event to the ledger.
2. The ledger receives and delivers the event to Azure Blockchain Workbench.
3. Azure Blockchain Workbench is subscribed to events from the ledger and receives the event.
4. Azure Blockchain Workbench publishes the event to subscribers on the Event Grid.
5. External systems are subscribed to the Event Grid, consume the message and take the appropriate action.

External System to a Smart Contract

There's also a scenario that flows from the opposite direction. In this case, an event is generated by a sensor or an external system and the data from that event should be sent to a smart contract. A common example is the delivery of data from financial markets, such as prices of commodities, stocks or bonds, to a smart contract. This occurs using the process depicted in **Figure 9**, where:

1. An event occurs in an external system that triggers the creation of a message for Azure Blockchain Workbench.
2. The external system has code written to create this message in a known format and sends this directly to the Service Bus.
3. Azure Blockchain Workbench is subscribed to events from the Service Bus and retrieves the message.
4. Azure Blockchain Workbench initiates a call to the ledger, sending data from the external system to a specific contract.
5. Upon receipt of the message, the contract transitions to a possible new state.

The implementation of an end-to-end solution for either integration scenario is beyond the scope of this article. Good examples of integration in both directions can be found at bit.ly/2M8yfiL.

In summary, similar integration patterns can be found in micro-service architectures and blockchain smart contracts. The distributed nature of both architectures encourages asynchronous messaging and the use of a message broker, in the form of an event grid or service

bus, for conveying information among services and smart contracts. Design patterns like event sourcing, ultimately, match the immutable nature of a digital ledger and the event-driven approach to off-chain communication. ■

STEFANO TEMPESTA is a Microsoft Regional Director and double MVP on AI and Business Applications, as well as chapter leader for the Microsoft Dynamics 365 community in Switzerland. Tempesta is an instructor of courses about Dynamics 365, blockchain and machine learning, and a regular speaker at international IT conferences, including Microsoft Ignite and Tech Summit. He founded Blogchain Space (blogchain.space), a blog about blockchain technologies, writes for MSDN Magazine and MS Dynamics World, and publishes machine learning experiments on the Azure AI Gallery (gallery.azure.ai).

THANKS to the following technical expert:
Jonathan Waldman

Imaging SDK for Winforms, WPF, and Web Development

GdPicture.NET



- ✦ Scanning
- ✦ OCR
- ✦ 100+ Formats
- ✦ Image Cleanup
- ✦ Annotations
- ✦ Barcodes
- ✦ Document Compression
- ✦ MICR
- ✦ Form Processing
- ✦ Thumbnails
- ✦ Viewer Control
- ✦ PDF
- ✦ Bookmarks
- ✦ Image Processing
- ✦ Color Detection
- ✦ Printing
- ✦ DICOM
- ✦ TIFF
- ✦ DOCX
- ✦ Metadata Support
- ✦ Document Conversion

Leverage your apps. with GdPicture.NET Imaging Toolkit

**60-day Free Trial
Support Included**

www.gdpicture.com



Visual Studio **LIVE!**

EXPERT SOLUTIONS FOR .NET DEVELOPERS

October 7-11, 2018
Hilton Resort

San Diego

Visual Studio **LIVE!** **25**
YEARS OF CODING INNOVATION

Code With Us in Sunny San Diego!

For the FIRST TIME EVER in our 25 year history, Visual Studio Live! is heading to San Diego, CA for up to 5 days of practical, unbiased, Developer training, including NEW intense hands-on labs.

Join us as we dig into the latest features of Visual Studio 2017, ASP.NET Core, Angular, Xamarin, UWP and more. Code with industry experts, get practical answers to your current challenges, and immerse yourself in the Microsoft platform. Plus, help us celebrate 25 years of coding innovation and experience the education, knowledge-share and networking at #VSLive25.



DEVELOPMENT TOPICS INCLUDE:



**DevOps in the
Spotlight**



**Cloud, Containers
and Microservices**



**AI, Data and
Machine Learning**



**Developing New
Experiences**



**Delivery and
Deployment**



**.NET Core
and More**



**Full Stack
Web Development**



Hands-On Labs



vslive.com/sandiego

**Best Savings End
September 14**

**Register by September 14
and Save \$200**

Use Promo Code MSDN

GOLD SPONSOR



SUPPORTED BY



PRODUCED BY



DevOps in the Spotlight		Cloud, Containers and Microservices		AI, Data and Machine Learning		Developing New Experiences		Delivery and Deployment		.NET and More		Full Stack Web Development	
START TIME	END TIME	Pre-Conference Full Day Hands-On Labs: Sunday, October 7, 2018 <small>(Separate entry fee required)</small>											
7:00 AM	8:00 AM	Hands-On Lab Registration - Coffee and Morning Pastries											
8:00 AM	5:00 PM	HOL01 Full Day Hands-On Lab: Xamarin and Azure: Build the Mobile Apps of Tomorrow - <i>Laurent Bugnion & Matthew Soucoup</i>						HOL02 Full Day Hands-On Lab: Building a Modern DevOps Pipeline with ASP.NET and Visual Studio Team Services - <i>Brian Randell</i>					
START TIME	END TIME	Pre-Conference Workshops: Monday, October 8, 2018 <small>(Separate entry fee required)</small>											
8:00 AM	9:00 AM	Pre-Conference Workshop Registration - Coffee and Morning Pastries											
9:00 AM	6:00 PM	M01 Workshop: Modern Security Architecture for ASP.NET Core - <i>Brock Allen</i>				M02 Workshop: SQL Server for Developers - <i>Andrew Brust and Leonard Lobel</i>				M03 Workshop: Distributed Cross-Platform Application Architecture - <i>Rockford Lhotka and Jason Bock</i>			
6:45 PM	9:00 PM	Dine-A-Round											
START TIME	END TIME	Day 1: Tuesday, October 9, 2018											
7:00 AM	8:00 AM	Registration - Coffee and Morning Pastries											
8:00 AM	9:15 AM	T01 An Introduction to TypeScript - <i>Jason Bock</i>			T02 Creating Four Beautiful Apps At Once with Xamarin.Forms - <i>Matthew Soucoup</i>			T03 Building Modern Web Apps with Azure - <i>Eric D. Boyd</i>			T04 DevOps on the Microsoft Stack with VSTS and TFS - <i>Brian Randell</i>		
9:30 AM	10:45 AM	T05 ASP.NET Core 2 for Mere Mortals - <i>Philip Japikse</i>			T06 Building Cross Device Experiences with Project Rome - <i>Tony Champion</i>			T07 Introduction to Azure Cosmos DB - <i>Leonard Lobel</i>			T08 Azure DevOps with VSTS, Docker, and K8 - <i>Brian Randell</i>		
11:00 AM	12:00 PM	KEYNOTE: To Be Announced											
12:00 PM	1:00 PM	Lunch											
1:00 PM	1:30 PM	Dessert Break - Visit Exhibitors											
1:30 PM	2:45 PM	T09 Assembling the Web—A Tour of WebAssembly - <i>Jason Bock</i>			T10 Lessons Learned from Making Resilient Apps with Azure Mobile App Services - <i>Matthew Soucoup</i>			T11 Modern SQL Server Security Features for Developers - <i>Leonard Lobel</i>			T12 Essential C# 8.0 - <i>Mark Michaelis</i>		
3:00 PM	4:15 PM	T13 JavaScript for the C# (and Java) Developer - <i>Philip Japikse</i>			T14 Enhancing UWP Experiences with Fluent Design - <i>Tony Champion</i>			T15 Predicting the Future Using Azure Machine Learning - <i>Eric D. Boyd</i>			T16 Getting to the Core of .NET Core - <i>Adam Tuliper</i>		
4:15 PM	5:30 PM	Welcome Reception											
START TIME	END TIME	Day 2: Wednesday, October 10, 2018											
7:30 AM	8:00 AM	Registration - Coffee and Morning Pastries											
8:00 AM	9:15 AM	W01 Angular 101 - <i>Deborah Kurata</i>			W02 Flying High with Xamarin! - <i>Sam Basu</i>			W03 Developing for Azure Using Visual Studio Code - <i>Brady Gaster</i>			W04 Monitor your Applications and Infrastructure - <i>Eric Boyd</i>		
9:30 AM	10:45 AM	W05 Introduction to OpenID Connect and OAuth2 for ASP.NET CORE - <i>Brock Allen</i>			W06 Cross-Platform Development with Xamarin, C#, and CSLA .NET - <i>Rockford Lhotka</i>			W07 Docker Containers on Azure—Let Me Count the Ways - <i>Michele Leroux Bustamante</i>			W08 DevOps for the SQL Server Database - <i>Brian Randell</i>		
11:00 AM	12:00 PM	GENERAL SESSION: 25 Years of VSLive!, 25 Years of Software - <i>Laurent Bugnion, Senior Global Azure Advocate, Microsoft</i>											
12:00 PM	1:00 PM	Birds-of-a-Feather Lunch											
1:00 PM	1:30 PM	Dessert Break - Visit Exhibitors - Exhibitor Raffle @ 1:15pm (Must be present to win)											
1:30 PM	2:45 PM	W09 N Things You Didn't Know About the Router - <i>Deborah Kurata</i>			W10 Essential Tools for Xamarin Developers! - <i>Sam Basu</i>			W11 Building Business Applications Using Bots - <i>Michael Washington</i>			W12 Get Good at DevOps: Feature Flag Deployments with ASP.NET & Web API - <i>Benjamin Day</i>		
3:00 PM	4:15 PM	W13 Securing Web APIs From JavaScript/Spa Applications - <i>Brock Allen</i>			W14 Azure, Windows and Xamarin: Using the Cloud to Power Your Cross-platform Applications - <i>Laurent Bugnion</i>			W15 Message-Based Microservices Architectures—Driven by Docker - <i>Michele Leroux Bustamante</i>			W16 Introducing Automated Testing into Legacy Code - <i>David Corbin</i>		
4:30 PM	5:45 PM	W17 Securing Web APIs from Mobile and Native Applications - <i>Brock Allen</i>			W18 Build Awesome AF Apps! - <i>Rachel Appel</i>			W19 Using The Microsoft Cognitive Custom Vision Service - <i>Michael Washington</i>			W20 Is Minimal Really Sufficient? A Look at Work Planning, Execution and Analytics - <i>David Corbin</i>		
7:00 PM	10:30 PM	Out On The Town - Explore the Downtown Gaslamp District!											
START TIME	END TIME	Day 3: Thursday, October 11, 2018											
7:30 AM	8:00 AM	Registration - Coffee and Morning Pastries											
8:00 AM	9:15 AM	TH01 Build Data Driven Web Apps using ASP.NET Core - <i>Rachel Appel</i>			TH02 Modernizing the Enterprise Desktop Application - <i>Oren Novotny</i>			TH03 Coaching Skills for Scrum Masters & The Self-Organizing Team - <i>Benjamin Day</i>			TH04 C# 7, Roslyn and You - <i>Jim Wooley</i>		
9:30 AM	10:45 AM	TH05 Tools for Modern Web Development - <i>Ben Hoelting</i>			TH06 An Architect's Guide to Data Science - <i>Becky Isserman</i>			TH07 Signing Your Code the Easy Way - <i>Oren Novotny</i>			TH08 Sharing C# Code Across Platforms - <i>Rockford Lhotka</i>		
11:00 AM	12:15 PM	TH09 Cryptography 101 - <i>Robert Boedigheimer</i>			TH10 Knockout: R vs Python for Data Science - <i>Becky Isserman</i>			TH11 Unit Testing & Test-Driven Development (TDD) for Mere Mortals - <i>Benjamin Day</i>			TH12 Improving Code Quality with Static Analyzers - <i>Jim Wooley</i>		
12:15 PM	1:15 PM	Lunch											
1:15 PM	2:30 PM	TH13 MVVM and ASP.NET Core Razor Pages - <i>Ben Hoelting</i>			TH14 Analytics and AI with Azure Databricks - <i>Andrew Brust</i>			TH15 How to Interview a Developer - <i>Billy Hollis</i>			TH16 Unit Testing Client-Side Applications - <i>Allen Conway</i>		
2:45 PM	4:00 PM	TH17 HTTP/2: What You Need to Know - <i>Robert Boedigheimer</i>			TH18 Power BI: What Have You Done for Me Lately? - <i>Andrew Brust</i>			TH19 The Most Important Lessons I've Learned in Forty Years of Developing Software - <i>Billy Hollis</i>			TH20 Migrating from AngularJS to Angular + TypeScript - <i>Allen Conway</i>		

Speakers and sessions subject to change

CONNECT WITH US



twitter.com/vslive – @VSLive



facebook.com – Search "VSLive"



linkedin.com – Join the "Visual Studio Live" group!

vslive.com/sandiego

Xamarin Productivity and Platform Improvements for Visual Studio 2017

Alessandro Del Sole

Mobile app development with Xamarin and Visual Studio, both on Windows and on the Mac, is crucial for Microsoft, which has made large investments to improve the quality and power of its mobile development tools and platforms. At the Build 2018 conference in May Microsoft unveiled a new generation of tools for cross-platform app development with C#, with a focus on productivity and performance. These include updates in the Xamarin tools for Visual Studio 2017, new features and improvements in Visual Studio for Mac, and the release of Xamarin.Forms 3.0.

In this article, I describe what's new for mobile app developers working with C# and Xamarin in Visual Studio 2017, walking through the features and updates that solve issues that existed in previous versions. You need Visual Studio 2017 version 15.7.3 or newer to use the new features, but I recommend that you install the latest available version (15.7.4 at this writing). I'll start with productivity updates and additions to Visual Studio 2017, and then I'll describe what's new in the platform support for iOS and Android.

This article discusses:

- XAML editor updates in Visual Studio 2017 that improve IntelliSense with powerful shortcuts, hints and other features
- Better platform support for iOS and Android in the latest version of Xamarin
- New features in Xamarin.Forms 3.0 and 3.1, including FlexLayout and the Visual State Manager

Technologies discussed:

Xamarin.Forms, Visual Studio 2017, iOS, Android

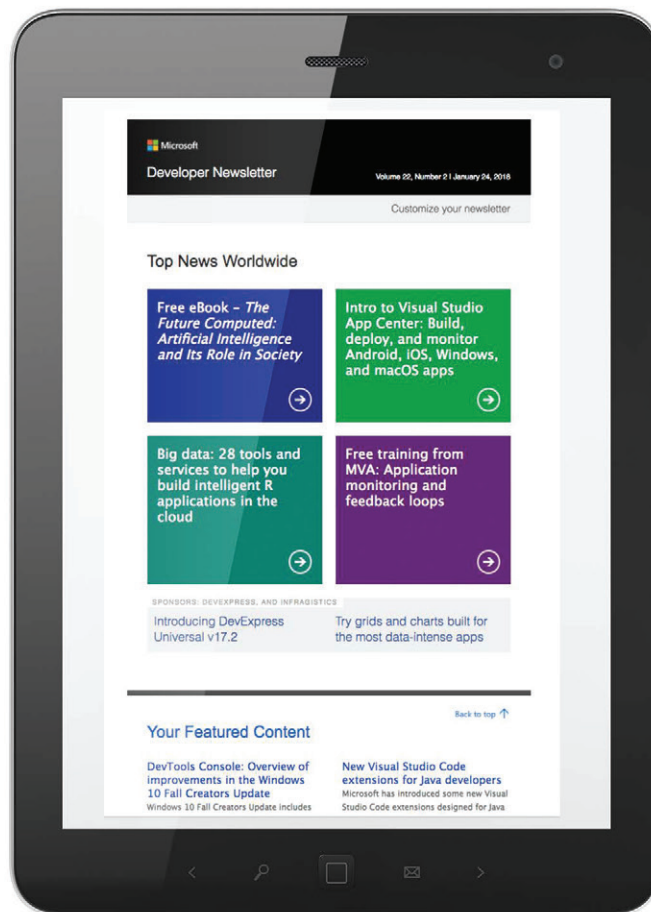
Visual Studio 2017 Tools for Xamarin

The Visual Studio 2017 IDE has recently been improved with new and updated tools for Xamarin, including XAML editor updates, integrated tools for iOS and Android updates, enhanced platform support, and new versions of Xamarin.Forms. These improvements are extremely important and deserve a thorough discussion, so the first part of this article walks through features that are related to both the coding experience and the integrated platform support.

XAML Editor Updates Developers working with Xamarin.Forms will benefit from the new XAML code editor, which is now powered by the same engine that drives Windows Presentation Foundation (WPF) and Universal Windows Platform (UWP). This is extremely important for several reasons:

- **Full, rich IntelliSense support:** Earlier versions of IntelliSense could make writing XAML markup painful to do. The new version adds linting, fuzzy matching and substring matching, which I'll describe in more detail.
- **Enhanced support to binding expressions:** IntelliSense now lists available objects for bindings based on the `{Binding}` markup extension and also lists available resources when using the `{StaticResource}` markup extension.
- **Quick actions and refactorings:** Light bulb suggestions are now available to XAML in Xamarin.Forms, just as they have been for WPF and UWP, making it easy to resolve XML namespaces, remove unused namespaces, and organize code with contextualized suggestions.

The feature known as fuzzy matching provides an algorithm that helps you find appropriate completions based on what you typed. For example, if you type `Stk` and then press `Tab`, IntelliSense will



Get news from MSDN in your inbox!

Sign up to receive the
MICROSOFT DEVELOPER NEWSLETTER,
which delivers the latest resources, SDKs,
downloads, partner offers, security news, and
updates on national and local developer events.

msdn
magazine

msdn.microsoft.com/flashnewsletter

add a StackLayout tag. Substring matching provides a list of possible completions as you type in a control name. For example, if you type Layout, IntelliSense will offer StackLayout, AbsoluteLayout and RelativeLayout as possible completions, as well as closing tags based on the same typing (see **Figure 1**). This feature is particularly useful when you only remember a control name partially.

Another interesting feature about string matching is CamelCase matching, which provides shortcuts based on CamelCase types. For instance, if you type RL and then press Tab, the editor inserts a RelativeLayout tag. With linting, the code editor immediately identifies some code issues, such as when a property has been specified more than once.

Many readers will know how useful features like light bulbs, quick actions and suggestions are, from their experience with both the C# code editor and the XAML code editor in WPF and UWP. Now these functions are available to Xamarin.Forms. For instance, if a code issue is detected, you can click the light bulb icon (or press Ctrl+.) and Visual Studio will show potential fixes, as you can see in **Figure 2**, where the code editor detects a missing XML namespace and suggests adding the proper namespace declaration.

You can use the light bulb and quick actions to remove unused namespaces and sort namespaces, which helps keep your code cleaner. IntelliSense also has full support now for binding expressions with markup extensions. For example, if you had a Person-ViewModel class you want to use as the binding context for your UI, and that's declared as a local resource, IntelliSense will then help you create the binding expression by showing the list of available resources.

With a binding context set up, IntelliSense can help with creating binding expressions, by showing a list of properties exposed by the bound object. **Figure 3** provides an example where you can also see a property called FullName, defined in the view model.

Beyond binding are improvements to popular existing features like Peek Definition and Go To Definition, which are now much more reliable in the Xamarin.Forms XAML editor. In fact, both features were problematic

before the latest Visual Studio updates. Now you can quickly navigate to the definition of an object or resource defined in XAML or C# code, eliminating the chore of manually browsing your code. Peek Definition opens an interactive popup directly in the active editor window, while Go To Definition opens a separate editor window pointing to the desired resource.

Xamarin.iOS Remote Update The XAML editing experience isn't the only area where Microsoft has worked to increase productivity. The tooling for Xamarin.iOS has also been enhanced.

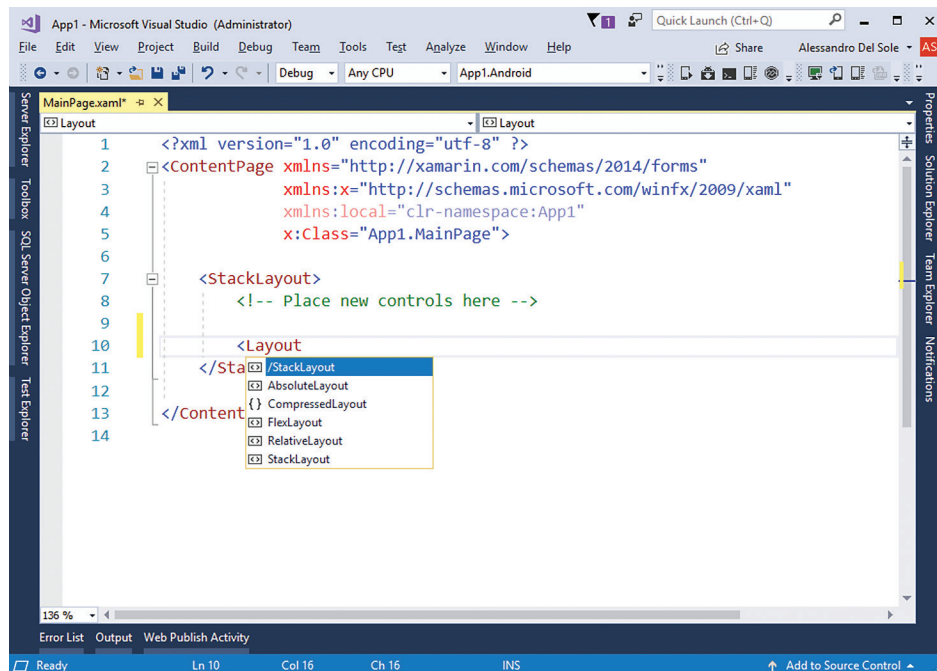


Figure 1 Substring Matching in the XAML Editor for Xamarin.Forms

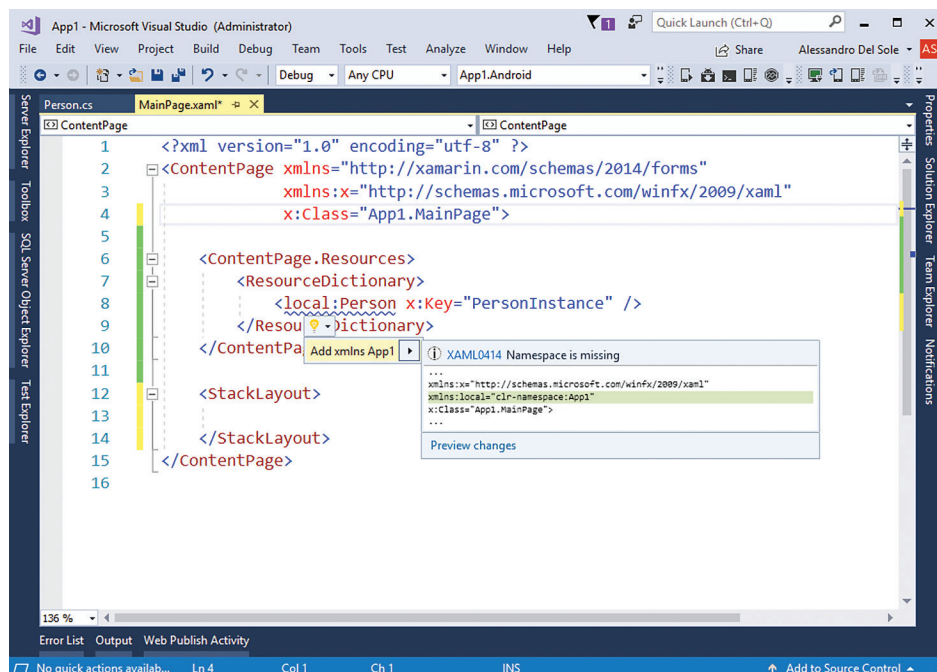


Figure 2 The Light Bulb Provides In-Line Quick Actions for Code Issues

Office & SharePoint LIVE!

ON-PREMISE, CLOUD & CROSS-PLATFORM TRAINING

ROYAL PACIFIC RESORT AT
UNIVERSAL ORLANDO
December 2-7, 2018



Training to Collaborate Anywhere

People are expected to work from anywhere at any time. Office & SharePoint Live! provides leading-edge knowledge and training to administrators and developers who must customize, deploy and maintain SharePoint Server on-premises and in Office 365 to maximize the business value. In addition, this event will also cover Office 365 and Office clients for developers & administrators. **Office & SharePoint Live! brings together Managers, IT Pros, DBAs and Developers for 6 days of workshops, keynotes and sessions to help you work through your most pressing projects.**

TRACK TOPICS INCLUDE:

- Managing Office 365
- SharePoint, Office 365 and the Cloud
- High-Value SharePoint Workloads: Search, Flow and PowerApps
- Developing for Office, Office 365 and SharePoint
- Office 365 Identity and Security

REGISTER
NOW

SAVE \$500 WITH SUMMER SAVINGS!
REGISTER BY AUGUST 31

Use Promo Code MSDN

A Part of Live! 360: The Ultimate Education Destination

6 CONFERENCES, 1 GREAT PRICE

Visual Studio LIVE!

SQL Server LIVE!

TECHMENTOR

Artificial Intelligence LIVE!

Office & SharePoint LIVE!

Modern Apps LIVE!



SQLLIVE360.COM

EVENT PARTNERS



SILVER SPONSOR



PREMIER & ALLIANCE MEDIA PARTNERS



SUPPORTED BY



PRODUCED BY



As you know, building apps for iOS with Xamarin in Visual Studio 2017 requires that the same version of the Xamarin.iOS SDK be installed on both your Windows PC and Mac machine. If there was a version mismatch, Visual Studio reported a warning and you had to update your Mac manually. This is no longer necessary. Visual Studio 2017 now offers to install the latest version of the Xamarin.iOS SDK on the Mac on your behalf. You simply click Install in the Pair to Mac dialog box and wait for the update to complete.

Beyond binding are improvements to popular existing features like Peek Definition and Go To Definition, which are now much more reliable in the Xamarin.Forms XAML editor.

Xamarin Tools for Automatic iOS Provisioning Creating apps for iOS requires some preliminary steps that can be really complicated. In addition to enrolling in Apple's Developer Program, which is required to enable your Apple developer account to sign and publish apps to the App Store, you need to provision your iOS device. iOS provisioning will enable you to deploy and test your app on a device. iOS provisioning is usually done via Xcode on the Mac. Put succinctly, iOS provisioning requires you to:

- **Create a development team:** This includes specifying the list of Apple developer accounts in the development team, enabling them to sign and publish an app.
- **Set up provisioning profiles:** A provisioning profile is bundled into the compiled app package and contains three things: an app unique identifier (App ID), one or more development certificates required to test an app on physical devices, and a Unique Device Identifiers (UDI) list that enumerates devices allowed to run an app.
- **Create signing certificates:** All apps must be signed before they can run on an iOS device, even for development, so a signing certificate is required. Different kinds of certificates are available to each developer account (such as development and publishing) depending on the level of subscription.

The Xamarin Team at Microsoft has created a useful document that explains how to get started with iOS provisioning (bit.ly/2Nhz000), but the reality is that complexity is the biggest barrier for developers wanting to build apps for iOS. Fortunately, the latest releases of Visual Studio 2017 include support for automatic iOS provisioning. You simply provide your Apple developer account and Visual Studio 2017 will set up all the necessary artifacts on your behalf, through a connection to a Mac machine. To accomplish this, you must first associate your Apple ID to Visual Studio via Tools | Options | Xamarin | Apple Accounts. In the Options dialog, you'll be able to associate one or more Apple IDs by clicking the Add button, after which Visual Studio 2017 shows the list of associated teams and the user role for each team.

Once you click OK, you can go to the iOS Bundle Signing tab of the Xamarin.iOS project properties and select the Automatic Provisioning option (see **Figure 4**).

You'll simply need to select the Team you wish to use for development from the Team dropdown and Visual Studio will generate the necessary provisioning profiles and signing certificates required to test apps on a physical device (which must be connected to the Mac machine).

Platform Support Improvements

The latest Xamarin releases include improvements that target several platforms, including Android, iOS, Tizen and macOS. In this article, I'll explore platform improvements for iOS and Android, both of which share an updated project template wizard. When you create a new iOS or Android project, you're prompted with a list of common templates (single view, master-detail, tabbed app and blank app). The project template wizard will also allow you to specify what kind of devices you want to support in your apps. Additionally, support for both platforms has been enhanced with specific improvements.

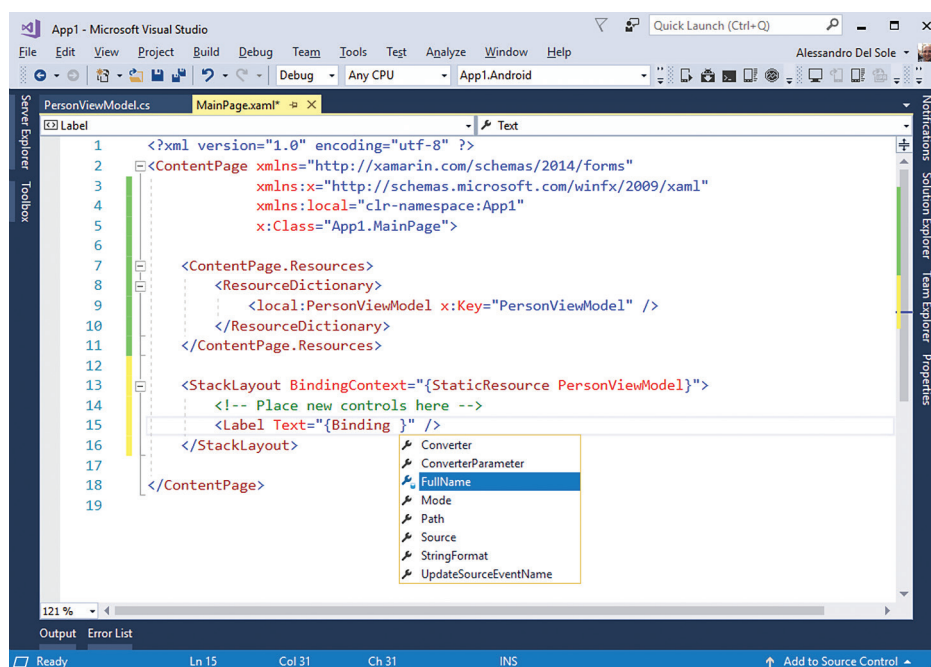


Figure 3 IntelliSense Makes It Easier to Create Binding Expressions

Spreadsheets Everywhere.



SpreadsheetGear 2017 Released

SpreadsheetGear 2017 adds a new SpreadsheetGear for .NET Standard product, official support for Excel 2013 and Excel 2016, 51 new Excel functions for a total of 449 fully supported functions, full conditional formatting support, enhanced workbook protection and encryption, cell gradient rendering and more.



Support for iOS, Android, Linux, macOS, UWP and more

SpreadsheetGear for .NET Standard enables cross-platform developers to enjoy the same high performance Excel-compatible reporting, charting, calculations and more relied on by thousands of Windows developers for 10+ years.



Scalable Reporting

Easily create richly formatted Excel reports without Excel from any ASP.NET, Windows Forms, WPF or Silverlight application using spreadsheet technology built from the ground up for performance, scalability and reliability.



Windows
Forms



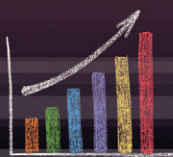
Silverlight



WPF

Powerful Controls

Add powerful Excel-compatible viewing, editing, formatting, calculating, filtering, sorting, charting, printing and more to your WinForms, WPF and Silverlight applications.



Comprehensive Charting

Enable users to visualize data with comprehensive Excel-compatible charting which makes creating, modifying, rendering and interacting with complex charts easier than ever before.



Fastest Calculations

Evaluate complex Excel-based models and business rules with the fastest and most complete Excel-compatible calculation engine available.

Download your free fully functional evaluation at SpreadsheetGear.com



SpreadsheetGear

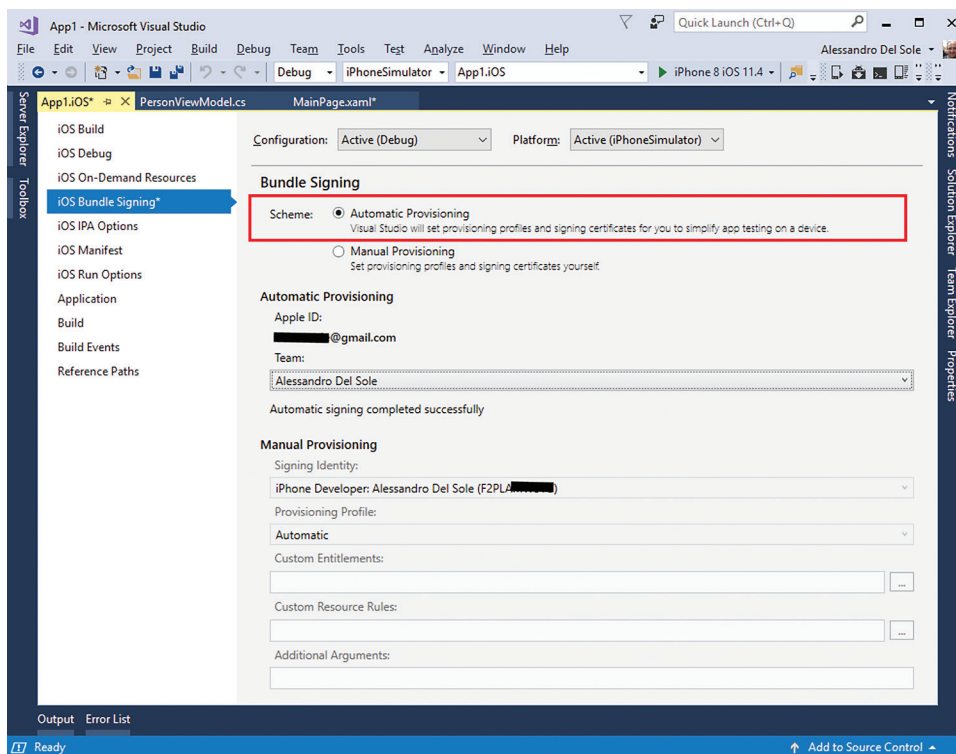


Figure 4 Enabling Automatic Provisioning

iOS Platform Support Xamarin.iOS has been refactored to include a fully static type system. As a result, iOS, macOS, tvOS and watchOS apps that you write with Xamarin are now between 30 percent and 50 percent smaller, because the linker is able to remove more code at build time. Memory usage has also decreased, especially at app startup. In addition, app startup is now much faster because most of the type registration is now done at build time rather than at runtime.

Android Platform Support Xamarin support for Android now includes a new tool called Android Device Manager, which ships with Visual Studio 2017 and makes it easier to create and maintain Android emulator images. You can launch the Android Device Manager via Tools | Android | Android Device Manager.

Not only can you create and edit emulator images, you can also quickly start an emulator instance via the Start button. This tool offers a large number of options that are out of the scope of this article, so refer to the documentation at bit.ly/2LhFUB1 for more information. Included in the recent update is the release of the Android Oreo SDK (API Level 27) and a feature known as Auto Install Android SDKs. If you open and build a project that requires a version of the Android SDK that you don't have on your machine, a notification will appear to help you download the required version in the background. This feature is disabled by default, so you have to open Tools | Options | Xamarin | Android Settings and select Enable Auto Install Android SDKs.

It's worth mentioning that Microsoft has also announced a preview of a Google Android emulator that runs on Hyper-V. This is a very important addition, because many developers used to work with the Visual Studio Android Emulator, which is based on Hyper-V and was recently discontinued by Microsoft. The new preview requires the

Windows 10 April 2018 update with the Windows HyperVisor Platform installed and Visual Studio 2017 15.8 Preview 1 (or higher). Because the emulator is still in preview, I don't cover it in this article, but you can read a very detailed and interesting blog post by Miguel de Icaza about it at bit.ly/2JsVrcq.

Xamarin.Forms Updates

Microsoft released Xamarin.Forms 3.0 at the Build 2018 conference in May, and more recently made Xamarin.Forms 3.1 available via NuGet. For developers building cross-platform apps with C#, the updates bring amazing new features and significant platform and UI improvements.

On the platform side, performance got a major boost in the form of improved fast renderers, which were introduced with version 2.5 last year. Renderers enable Xamarin.Forms to translate visual

elements into native views—work that can get pretty expensive, especially with complex views. Microsoft redesigned the way renderers work in version 3.0 to boost rendering speed and performance.

On the UI side, Xamarin.Forms 3.0 introduces a new layout called FlexLayout. It works like a StackLayout, arranging child visual elements vertically or horizontally, while also being able to wrap the child visual elements if there's not enough space in a single row or column. The following code snippet shows an example of how you can easily use a FlexLayout:

```
<FlexLayout Wrap="Wrap" JustifyContent="SpaceAround">
  <!-- Place new controls here -->
  <Label Text="A label" FlexLayout.AlignSelf="Center"/>
  <Image Source="Image1.jpg" FlexLayout.AlignSelf="Center"/>
  <Button x:Name="Button1" Text="Tap here" FlexLayout.AlignSelf="Center"/>
</FlexLayout>
```

There are numerous FlexLayout properties, but the three most important are:

- **Wrap:** A value from the FlexWrap enumeration that specifies if FlexLayout content should be wrapped to the next row if there's not enough space in the first. Possible values are Wrap (wraps to the next row), NoWrap (keeps the view content on one row) and Reverse (wraps to the next row in the opposite direction).
- **Direction:** A value from the FlexDirection enumeration that determines if the children of the FlexLayout should be arranged in a single row or column. The default value is Row, but other possible values are Column, RowReverse and ColumnReverse (where Reverse means that child views will be laid out in the reverse order).
- **JustifyContent:** A value from the FlexJustify enumeration that specifies how child views should be arranged when there's

extra space around them. There are self-explanatory values such as Start, Center and End, as well as more granular options. These include SpaceAround, where elements are spaced with one unit of space at the beginning and end, and two units of space between them, so the elements and the space fill the line, and SpaceBetween, where child elements are spaced with equal space between units and no space at either end of the line, again so the elements and the space fill the line. The SpaceEvenly value causes child elements to be spaced so the same amount of space is set between each element as there is from the edges of the parent to the beginning and end elements.

CSS styles in Xamarin.Forms can be declared and consumed in different ways.

Within child views, you can set the `FlexLayout.AlignSelf` attached property to specify how the view should be aligned inside the axis of the `FlexLayout`. This is an object of type `FlexAlignItems` and possible values are Auto, Start, Center, End and Stretch. The complete documentation for the `FlexLayout` control is available at bit.ly/2LaMzgt. Another interesting addition is support for Cascading Style Sheets (CSS), though it's worth mentioning that CSS styles must be compliant

to `Xamarin.Forms` to be consumed in mobile apps (see bit.ly/2miMcS0) and that not all CSS styles are supported. Therefore, this feature should be considered a complement to XAML styles, rather than a replacement. Also, CSS styles are parsed and evaluated at runtime, not at build time.

CSS styles in `Xamarin.Forms` can be declared and consumed in different ways. One option is within a `StyleSheet` tag to be added as a resource in a page, like in the following snippet:

```
<ContentPage.Resources>
  <ResourceDictionary>
    <StyleSheet>
      <![CDATA[
        ^contentpage {
          background-color: lightgray; }
        stacklayout {
          margin: 20; }
      ]]>
    </StyleSheet>
  </ResourceDictionary>
</ContentPage.Resources>
```

The content of the CSS is enclosed in a CDATA section. For each visual element, you provide the desired property/value pairs. Another option is to consume an existing style from a .css file, as follows:

```
<ContentPage.Resources>
  <ResourceDictionary>
    <StyleSheet Source="/Assets/mystyle.css" />
  </ResourceDictionary>
</ContentPage.Resources>
```

Remember that the .css file must be bundled with your application by setting its Build Action property as `EmbeddedResource`. CSS styles can also be declared and consumed in C# code. The following snippet shows how to declare and add a style to a page's resources:



SUPER-FAST AND ADVANCED CHARTS

LightningChart®

- WPF and WinForms
- Real-time scrolling up to 2 billion points in 2D
- Hundreds of examples
- On-line and off-line maps
- Advanced Polar and Smith charts
- Outstanding customer support



2D charts - 3D charts - Maps - Volume rendering - Gauges
www.LightningChart.com/ms

**TRY FOR
FREE**



```
using (var reader =
    new StringReader
    ("^contentpage { background-color: lightgray; } stacklayout { margin: 20; }"))
{
    this.Resources.Add(StyleSheet.FromReader(reader));
}
```

You could even load a .css file at runtime using the following snippet, where `Page1` represents a page in the application and `Project1` represents your project name:

```
this.Resources.Add(StyleSheet.FromAssemblyResource(
    IntrospectionExtensions.GetTypeInfo(typeof(Page1)).Assembly,
    "Project1.Assets.mystyle.css"));
```

Another extremely useful addition to Xamarin.Forms is the Visual State Manager (bit.ly/2NVH4AS), which you might already know if you have experience with WPF and UWP. With it, you can make changes to the UI you declared in XAML, based on a view's state. For example, you can use the Visual State Manager to change the color of a view depending on its state, as shown in the code snippet in **Figure 5**.

With the markup in **Figure 5**, the Entry will automatically change its background color when its state changes. In this case, you need to set the Entry's `IsEnabled` property as `False` to disable the view and trigger the Disabled state. States must be grouped into objects called `VisualStateGroup`. Each state is represented by the `VisualState` object, where you add Setter specifications as you would do with styles, therefore providing the name of the property you want to change and its value. Of course, you can specify multiple property setters.

Xamarin.Forms defines a set of states called common states, such as Normal, Focused and Disabled (see the `VisualStateGroup` with the `CommonState` name above); these states are common to each view. Other states might be available only to specific views and not to others. The Visual State Manager provides an elegant and clean way to control the UI behavior, all in your XAML code.

Xamarin.Forms 3.0 also makes it easy to implement right-to-left localization. The `Device` class now exposes the `FlowDirection` property, which reads this localization information from the device. You can then assign its value to the `FlowDirection` property of a view as follows:

```
<ContentPage FlowDirection="{x:Static Device.FlowDirection}">
```

Of course you can also bind the `FlowDirection` property of a view in XAML, making sure your view model exposes a property that returns the value of `Device.FlowDirection`. Just a few weeks after the release of

Figure 5 Changing Background Color

```
<Entry>
  <VisualStateManager.VisualStateGroups>
    <VisualStateGroup x:Name="CommonStates">
      <VisualState x:Name="Normal">
        <VisualState.Setters>
          <Setter Property="BackgroundColor" Value="White" />
        </VisualState.Setters>
      </VisualState>

      <VisualState x:Name="Focused">
        <VisualState.Setters>
          <Setter Property="BackgroundColor" Value="LightGray" />
        </VisualState.Setters>
      </VisualState>

      <VisualState x:Name="Disabled">
        <VisualState.Setters>
          <Setter Property="BackgroundColor" Value="Gray" />
        </VisualState.Setters>
      </VisualState>
    </VisualStateGroup>
  </VisualStateManager.VisualStateGroups>
</Entry>
```

Xamarin.Forms 3.0, Microsoft published version 3.1, which provides bug fixes and introduces many improvements to performance, reliability and visual elements. For instance, the `SelectionMode` property is now available in the `ListView` control, which can be used as follows:

```
<ListView SelectionMode="None">
  ...
</ListView>
```

Possible values are `None` and `Single`. This is an important improvement, because previous versions required that you manually write code to disable an item selection or create a custom renderer for the `ListView`. The `Switch` control also now supports specifying a different color when the selector is turned on, which you can do with the `OnColor` property of type `Color`, like so:

```
<Switch OnColor="Blue"/>
```

Before Xamarin.Forms 3.1, you had to write a custom renderer to get this result. Similarly, the `Slider` control allows specifying colors through the `ThumbColor`, `MaximumTrackColor` and `MinimumTrackColor` properties. Other additions are the `IsSpellCheckEnabled` property for `Entry` and `Editor` views, and the control over text-prediction and APIs that allow you to control the shadow over views in iOS.

The addition of bindable spans is worth taking a moment to explore, given this feature had been requested many times by the developer community. Put succinctly, the `Span` class, which you use for more sophisticated string formatting, now inherits from `BindableObject`, so that all the properties related to text formatting support data binding. A new bindable property called `Style` has been introduced, so you can use XAML styles defined in the app resources. The following XAML snippet provides an example:

```
<Label>
  <Label.FormattedText>
    <FormattedString>
      <FormattedString.Spans>
        <Span Text="{Binding TitleText}" Style="{StaticResource TitleStyle}" />
        <Span Text="{Binding SubtitleText}" Style="{StaticResource SubTitleStyle}" />
        <Span Text="{Binding SomeText}" Style="{StaticResource NormalStyle}" />
      </FormattedString.Spans>
    </FormattedString>
  </Label.FormattedText>
</Label>
```

You can find a full list of available additions in Xamarin.Forms 3.1 in the release notes page at bit.ly/2NKVA3T.

Wrapping Up

Xamarin is continuously advancing, not only to support evolving platforms and OSes, but to increase the quality and productivity of the development tools. In this article I've explored a huge number of new and updated features that solve many problems, from the code editor and integrated tools to platform and CSS support. More is yet to come as additional updates are released. It's a good idea to regularly check the Xamarin blog (blog.xamarin.com) to stay up-to-date with the latest news and announcements. ■

ALESSANDRO DEL SOLE has been a Microsoft MVP since 2008 and a Xamarin Certified Developer. He has authored many books, eBooks, instructional videos and articles about .NET development with Visual Studio. Del Sole works for Fresenius Medical Care as a senior software engineer, focusing on building .NET and mobile apps in the health care market. You can follow him on Twitter: @progaalex.

THANKS to the following Microsoft technical experts who reviewed this article: David Britch, Amy Burns

Manipulating Documents?

APIs to view, convert, annotate, compare, sign, assemble
and search documents in your applications.

Try GroupDocs APIs for FREE

Download a Free Trial at

<https://downloads.groupdocs.com>

Microsoft
.NET



 GROUPDOCS



GroupDocs.Viewer

View over 50 documents and image formats in any application using document viewer APIs.



GroupDocs.Annotation

Add annotations to specific words, phrases and any region of the document.



GroupDocs.Conversion

Fast batch document conversion APIs for any .NET, Java or Cloud app.



GroupDocs.Comparison

Compare two documents and get a difference summary report.



GroupDocs.Signature

Digitally sign Microsoft Word, Excel, PowerPoint and PDF documents.



GroupDocs.Assembly

Document automation APIs to create reports from templates and various data sources.



GroupDocs.Metadata

Organize documents with metadata within any cross platform application.



GroupDocs.Search

Transform your document search process for advance full text search capability.

► GroupDocs.Text

► GroupDocs.Editor

► GroupDocs.Parser

► GroupDocs.Watermark

Americas: +1 903 306 1676

EMEA: +44 141 628 8900
sales@asposeptyltd.com

Oceania: +61 2 8006 6987



Visual Studio **LIVE!**

EXPERT SOLUTIONS FOR .NET DEVELOPERS

**ROYAL PACIFIC RESORT AT
UNIVERSAL ORLANDO**
December 2-7, 2018



Coding in Paradise

Developers, engineers, software architects and designers will code in paradise this December at Visual Studio Live! (VSLive!™). Help us celebrate 25 years of coding innovation by returning to warm, sunny Orlando for six days of unbiased and cutting-edge education on the Microsoft Platform. Soak in the knowledge on everything from Visual Studio and the .NET framework, to ASP.NET, .NET Core, JavaScript, Xamarin, Database Analytics, and so much more. VSLive!™ features 60+ sessions led by industry experts and Microsoft insiders.

Grab your flip flops, and your laptops, and make plans to attend the conference more developers rely on to expand their .NET skills and the ability to build better applications.

**REGISTER
NOW**

**SAVE \$400 WITH
SUPER EARLY BIRD PRICING!
REGISTER BY OCTOBER 5**

Use Promo Code MSDN

A Part of Live! 360: The Ultimate Education Destination

6 CONFERENCES, 1 GREAT PRICE

Visual Studio **LIVE!**

SQL Server **LIVE!**

TECHMENTOR

Artificial
Intelligence **LIVE!**

Office &
SharePoint **LIVE!**

ModernApps **LIVE!**



VSLIVE.COM/ORLANDO

EVENT PARTNERS



SILVER SPONSORS



EXHIBITORS



SUPPORTED BY



PRODUCED BY



AGENDA-AT-A-GLANCE

DevOps in the Spotlight		Cloud, Containers and Microservices		App Dev Data		Developing New Experiences		Delivery and Deployment		.NET and More		Full Stack Web Development	
START TIME	END TIME	Visual Studio Live! Full Day Hands-On Labs: Sunday, December 2, 2018											
9:00 AM	6:00 PM	VSS01 Hands-On Lab: Develop an ASP.NET Core 2 and EF Core 2 App in a Day - Philip Japikse				VSS02 Hands-On Lab: From 0-60 in a Day with Xamarin and Xamarin.Forms - Roy Cornelissen & Marcel de Vries				VSS03 Hands-On Lab: Busy Developer's Workshop on VueJS - Ted Neward			
2:00 PM	7:00 PM	Pre-Conference Registration - Royal Pacific Resort Conference Center											
START TIME	END TIME	Visual Studio Live! Pre-Conference Workshops: Monday, December 3, 2018											
8:30 AM	5:30 PM	VSM01 Workshop: Web Development in 2018 - Chris Klug				VSM02 Workshop: Architect and Build a Modern ASP.NET App in the Azure Cloud with a full CI/CD Pipeline with VSTS - Brian Randell and Miguel Castro				VSM03 Workshop: Cross-Platform C# Using .NET Core and WebAssembly - Rockford Lhotka & Jason Back			
6:30 PM	8:00 PM	Dine-A-Round Dinner @ Universal CityWalk - 6:30pm - Meet at Conference Registration Desk to walk over with the group											
START TIME	END TIME	Visual Studio Live! Day 1: Tuesday, December 4, 2018											
8:00 AM	9:00 AM	VISUAL STUDIO LIVE! & MODERN APPS LIVE! KEYNOTE: .NET Everywhere and for Everyone James Montemagno, Principal Program Manager - Mobile Developer Tools, Microsoft											
9:15 AM	10:30 AM	VST01 ASP.NET Core 2 For Mere Mortals - Philip Japikse			VST02 Busy Developer's Guide to Kotlin - Ted Neward			VST03 Azure 101 - Laurent Bugnion			VST04 Modernizing Your Source Control: Migrating to Git from Team Foundation Version Control (TFVC) - Colin Dembovsky		
10:30 AM	11:00 AM	Networking Break • Visit the EXPO - Pacifica 7											
11:00 AM	12:00 PM	LIVE! 360 KEYNOTE: To Be Announced - Pacifica 6											
12:00 PM	12:45 PM	Lunch • Visit the EXPO											
12:45 PM	1:30 PM	Dessert Break • Visit the EXPO											
1:30 PM	1:50 PM	VST05 Fast Focus: Serverless Computing: Azure Functions and Xamarin in 20 minutes - Laurent Bugnion				VST06 Fast Focus: Getting Git - Jason Back				VST07 Fast Focus: Xamarin.Essentials - Cross-Platform APIs for Your Mobile Apps - James Montemagno			
2:00 PM	2:20 PM	VST08 Fast Focus: Getting Started with ASP.NET Core 2.0 Razor Pages - Walt Ritscher				VST09 Fast Focus: Cross Platform Device Testing with xUnit - Oren Novotny				VST10 Fast Focus: Get Your Full .NET Code into .NET Standard - Rockford Lhotka			
2:20 PM	2:45 PM	Networking Break • Visit the EXPO - Pacifica 7											
2:45 PM	4:00 PM	VST11 Introduction to Webpack - Chris Klug			VST12 Xamarin.Forms Takes You Places! - Sam Basu			VST13 Azure, Windows and Xamarin: Using the Cloud to Power Your Cross-platform Applications - Laurent Bugnion			VST14 Writing Testable Code and Resolving Dependencies - DI Kills Two Birds with One Stone - Miguel Castro		
4:15 PM	5:30 PM	VST15 Angular Application Testing Outside the Church of TDD - Chris Klug			VST16 Busy .NET Developer's Guide to Python - Ted Neward			VST17 HoloLens, Mixed Reality & VR Development with the Cloud - Nick Landry			VST18 Testing in Production Using Azure and Visual Studio Team Services (VSTS) - Colin Dembovsky		
5:30 PM	7:30 PM	Exhibitor Reception - Pacifica 7											
START TIME	END TIME	Visual Studio Live! Day 2: Wednesday, December 5, 2018											
8:00 AM	9:15 AM	VSW01 The Whirlwind Tour of Authentication and Authorization with ASP.NET Core - Chris Klug			VSW02 Essential Tools for Xamarin Developers! - Sam Basu			VSW03 Introduction to Windows Containers and Docker - Marcel de Vries			VSW04 OWASP DevSlop: DevSecOps with VSTS & Azure - Tanya Janca		
9:30 AM	10:45 AM	VSW05 Assembling the Web - A Tour of WebAssembly - Jason Back			VSW06 Cross-Platform App Dev with Xamarin and CSLA .NET - Rockford Lhotka			VSW07 Microservices with AKS (Azure Kubernetes Service) - Vishwas Lele			VSW08 To Be Announced		
10:45 AM	11:30 AM	Networking Break • Visit the EXPO - Pacifica 7											
11:30 AM	12:30 PM	LIVE! 360 KEYNOTE: Enterprise Transformation (And You Can Too) Donovan Brown, Principal DevOps Manager, Cloud Developer Advocacy Team, Microsoft											
12:30 PM	1:30 PM	Birds-of-a-Feather Lunch											
1:30 PM	2:00 PM	Dessert Break • Visit the EXPO											
2:00 PM	3:15 PM	VSW09 No Strings Attached: JavaScript without Webpack, Transpilers, or Frameworks - Ashley Grant			VSW10 DevOps for the SQL Server Database - Brian Randell			VSW11 Programming with Microsoft Flow - Walt Ritscher			VSW12 What's New in C# 7 - Phil Japikse		
3:15 PM	4:00 PM	Networking Break • Visit the EXPO • Expo Raffle @ 3:30 p.m. - Pacifica 7											
4:00 PM	5:15 PM	VSW13 Electron: Desktop Development For Web - Chris Woodruff			VSW14 Entity Framework for Enterprise Applications - Benjamin Day			VSW15 The Mystical World of I/O Bindings in Azure Functions - Ashley Grant			VSW16 Signing Your Code the Easy Way - Oren Novotny		
7:30 PM	9:00 PM	Live! 360 Dessert Luau - Wantilan Pavilion											
START TIME	END TIME	Visual Studio Live! Day 3: Thursday, December 6, 2018											
8:00 AM	9:15 AM	VSH01 Encrypting the Web - Robert Boedigheimer			VSH02 Learning The Language Of HTTP For A Better Data Experience In Your Mobile Apps - Chris Woodruff			VSH03 How to Interview a Developer - Billy Hollis			VSH04 Creating a Release Pipeline with Team Services - Esteban Garcia		
9:30 AM	10:45 AM	VSH05 Advanced Fiddler Techniques - Robert Boedigheimer			VSH06 Making XAML Do Things You Didn't Realize It Could - Billy Hollis			VSH07 What Developers Want - Rabeb Othmani			VSH08 To Be Announced		
11:00 AM	12:00 PM	VISUAL STUDIO LIVE! PANEL DISCUSSION: What Matters Most for the future of Modern Apps: AI, Data, Security, or UX? Brian Randell (Moderater), Billy Hollis, Tanya Janca, Oren Novotny, & Rabeb Othmani											
12:00 PM	1:00 PM	Lunch on the Lanai - Lanai / Pacifica 7											
1:00 PM	2:15 PM	VSH09 Get Func-y: Understanding Delegates in .NET - Jeremy Clark			VSH10 Non-Useless Unit Testing Entity Framework & ASP.NET MVC - Benjamin Day			VSH11 Modernizing the Enterprise Desktop Application - Oren Novotny			VSH12 DevOps: A Catalyst for Enterprise Agility - Heidi Araya & Esteban Garcia		
2:30 PM	3:45 PM	VSH13 IEnumerable, ISaveable, IDontGetIt: Understanding .NET Interfaces - Jeremy Clark			VSH14 Finding Your Place in the Cosmos: When and Why You Should Consider Azure Cosmos DB - Eric Potter			VSH15 Pushing Left Like a Boss: Application Security Foundation - Tanya Janca			VSH16 Scrum, Kanban, or Scrumban? - Heidi Araya		
4:00 PM	5:00 PM	Next? Live! 360 Networking Event Heidi Araya, Andrew Brust, Jeremy Clark, Ben Curry, Benjamin Day, Peter De Tender, Brent Edwards, Kevin Ford, Grant Fritchey, Esteban Garcia, Seth Juarez, Dave Kawula, Crista Kawula, Sami Laiho, Thomas LaRock, Vishwas Lele, Rockford Lhotka, Karen Lopez, Matthew McDermott, Brian Randell, Paul Schaefflein, Jen Underwood, Rob Windsor											
START TIME	END TIME	Visual Studio Live! Post-Conference Workshops: Friday, December 7, 2018											
8:00 AM	05:00 PM	VSF01 Workshop: UX Design for Developers: Basics of Principles and Process - Billy Hollis						VSF02 Workshop: Microservices with AKS (Managed Kubernetes) - Vishwas Lele					

Speakers and sessions subject to change



Never Mind JavaScript, Here's Blazor

About a decade ago the mainstream Web stopped at a crossroad. Traditionally interpreted, client-side technologies such as HTML, CSS and JavaScript were about to disappear, progressively overwhelmed by compiled technologies and languages such as XAML and C#. At the same time, the browser environment seemed poised to incorporate a native, dedicated processing engine like Silverlight, which had been installed as a separate plug-in. Then the first drafts of HTML 5 appeared at the horizon and, shortly thereafter, Silverlight was sunsetted.

The turn of events shook up the landscape as IT decision makers sought a viable platform for cross-platform Web development. HTML5 received a boost and JavaScript began its comeback. Today, Angular, React and Vue are highly popular JavaScript-based frameworks that more and more development teams are choosing for building their front ends.

Blazor relies extensively on work done by Xamarin to bring its Mono framework to WASM.

Blazor comes up as a possible alternative to Angular/React/Vue and, thanks to WebAssembly (WASM), it opens up the Web to be able to take advantage of existing development ecosystems, including .NET. Therefore, more than just another Model-View-ViewModel (MVVM)-oriented JavaScript framework, Blazor is, to some extent, a modern attempt to revamp and refine the Silverlight concept—bringing the power of C# to client-side Web development. The core of Blazor is designed to provide a strongly typed, browser-based environment upon which to build Web front ends using C# and .NET Core, in lieu of JavaScript and Razor on top of HTML. However, Blazor doesn't push HTML and CSS to the corner. Rather, it extends the HTML syntax with a collection of predefined Razor tag helpers while requiring no plug-in to be installed.

I caution that Blazor is still largely in development and whatever you read here may radically change in only a few weeks.

Code download available at bit.ly/2LVCxA.

The code presented in this article is based on version 0.5.0 released in July 2018.

In this article, I'll discuss the underlying system architecture of a Blazor application and present a minimal "Hello World" application.

The Magic of Blazor

If you're coming from a Silverlight background, you might wonder how Blazor—a non JavaScript platform—can run in the browser without an installable plug-in. The answer is that, well, it can't. Blazor actually requires some sort of plug-in to host it within a compliant browser. The trick is that the so-called conceptual plug-in is already embedded in most modern browsers (those released in the past 12 months), and can be easily polyfilled via JavaScript in older browsers. Unlike discreet plug-ins like Adobe Flash or Silverlight, the tool enabling Blazor support is WASM, an open standard being developed in a W3C Community Group that includes representatives from all major browsers.

Technically, WASM is a binary format for a stack-based virtual machine (VM), architecturally similar to the .NET CLR and Java Virtual Machine, but unlike the Android Dalvik VM. As the compilation target of high-level programming languages, WASM has the potential to enable deployment of .NET applications over the Web. For this to happen, however, a bridge is needed to convert the .NET runtime to WASM. Blazor relies extensively on work done by Xamarin to bring its Mono framework to WASM. The .NET code contained in a Blazor application runs against a version of Mono specifically compiled to WASM, as shown in **Figure 1**.

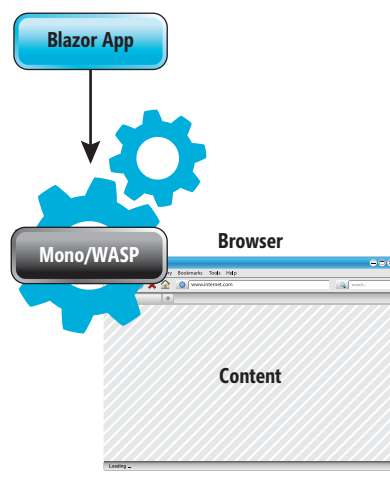


Figure 1. .NET Code Running in the Browser

How is this different from, say, Silverlight? Silverlight applications ran sandboxed in a browser-specific shell communicating with the container browser environment through ActiveX in Internet Explorer, and through NPAPI in other browsers like Chrome and Firefox (see documentation at bit.ly/2kDgdY). Today, most browsers are dismissing NPAPI support in favor of WebAssembly. Blazor takes many of the best concepts of Silverlight and applies them in a way that appeals to developers seeking an alternative to JavaScript.

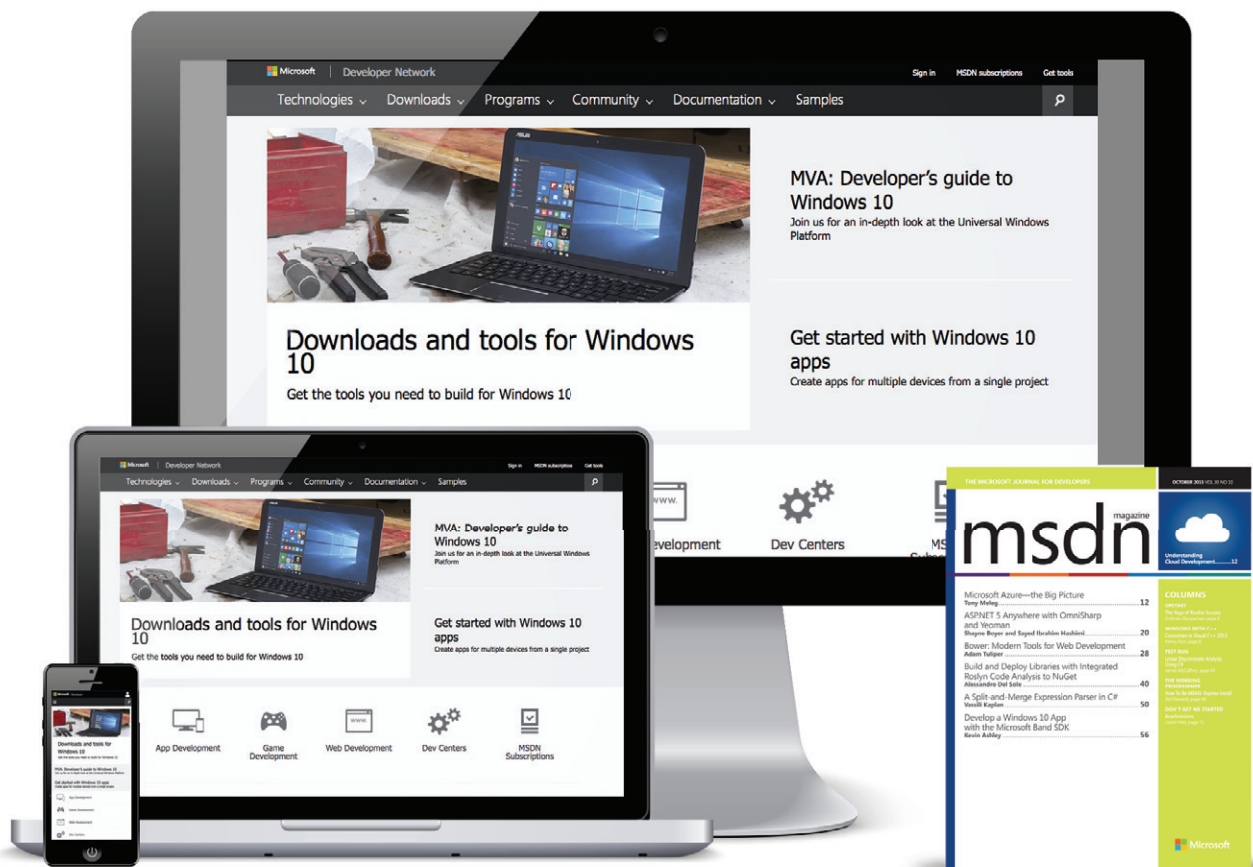
What about legacy browsers?

Development tools cross compile code to both WASM and asm.js, which is a low-level subset of JavaScript, intended to be a compile target from high-level languages (see asmjs.org). Blazor, in particular, falls back to using an

msdn

magazine

Where you need us most.



Visual Studio **LIVE!**
EXPERT SOLUTIONS FOR .NET DEVELOPERS

LIVE!
360
TECH EVENTS WITH PERSPECTIVE

MSDN.microsoft.com

asm.js-based .NET runtime. Then the deployed code performs feature detection in the browser to determine how to run. Note that the entire asm.js framework is currently under research and development.

You can visit caniuse.com to check which browser versions currently support WASM. The site notes that all browser versions (both desktop and mobile) released from mid-2017 and on support it. Note that at the time of this writing, Blazor polyfill was temporarily not working with Internet Explorer.

Anatomy of a Blazor Application

A Blazor application is a plain .NET Standard library with a Program.Main entry point that gets downloaded and run in the browser. Only the .NET runtime is

compiled to WASM; the source code you write in C# runs natively as if it were a normal .NET application. To clear security concerns, note that all Blazor code still runs in the same secure sandbox as JavaScript. **Figure 2** shows the list of files downloaded by the sample Blazor application being presented in the rest of this article.

The different background color you see in the figure identifies two distinct moments in the lifetime of a Blazor application. First, the blazor.js and mono.js files are downloaded to coordinate the download of the Mono WebAssembly runtime (the mono.wasm file). At present, the Mono runtime includes a relatively feature-rich version of .NET and weighs in at 600KB. However, in earlier versions a compact option was available at a fraction of the size. This is an area that could use some improvement and optimization.

Once the Mono runtime has been downloaded, it begins the download of the actual .NET assemblies. The initialization step may take a while—about three seconds in the example in **Figure 2**. For this reason, the standard Blazor Visual Studio template provides a dedicated area for the UI to show during the loading phase. Let's take a look at a sample application.

Building a Sample Application

To play with Blazor, you need .NET Core 2.1 and Visual Studio 15.7 or newer. In addition, you need to install the Blazor templates from the Visual Studio marketplace. The template is located under the ASP.NET Core Web Application folder. There are currently three templates—plain client-side Blazor, ASP.NET Core-hosted application and server-side Blazor. Let's go for a plain Blazor application with no dependencies on any back-end environment. This scenario resembles very closely that of an Angular/React or even Silverlight application—a plain client application downloaded and run within the browser.

As mentioned, the project in Visual Studio is a .NET Standard 2.0 console application based on two specific NuGet packages:

Name	Status	Type	Initiator	Size
localhost	200	document	Other	1.1 KB
bootstrap.min.css	200	stylesheet	(index)	27.4 KB
jquery-3.1.1.min.js	200	script	(index)	33.7 KB
bootstrap.bundle.min.js	200	script	(index)	23.9 KB
loading.gif	200	gif	(index)	17.5 KB
blazor.js	200	script	(index)	83.1 KB
mono.js	200	script	MonoPlatform.ts:193	50.9 KB
mono.wasm	200	fetch	mono.js:1	697 KB
favicon.ico	200	text/html	Other	1.1 KB
Blazor0.dll	200	xhr	MonoPlatform.ts:251	7.3 KB
Microsoft.AspNetCore.Blazor.Browser.dll	200	xhr	MonoPlatform.ts:251	24.7 KB
Microsoft.AspNetCore.Blazor.dll	200	xhr	MonoPlatform.ts:251	55.4 KB
Microsoft.Extensions.DependencyInjection.Abstractions.dll	200	xhr	MonoPlatform.ts:251	18.2 KB
Microsoft.Extensions.DependencyInjection.dll	200	xhr	MonoPlatform.ts:251	23.2 KB
mscorlib.dll	200	xhr	MonoPlatform.ts:251	656 KB
netstandard.dll	200	xhr	MonoPlatform.ts:251	9.5 KB
System.Core.dll	200	xhr	MonoPlatform.ts:251	138 KB
System.dll	200	xhr	MonoPlatform.ts:251	42.5 KB
System.Net.Http.dll	200	xhr	MonoPlatform.ts:251	31.4 KB
data:image/svg+xml;...	200	svg+xml	Other	(from me...

20 requests | 1.9 MB transferred | Finish: 4.54 s | DOMContentLoaded: 459 ms | Load: 495 ms

Figure 2 Files Downloaded by the Sample Blazor Application

Blazor.Build and Blazor.Browser. Both are rooted in the Microsoft.AspNetCore namespace.

The project is a standard .NET Core project with a Program.cs bootstrapper, a Startup.cs class, and a few folders that recall an ASP.NET Core application, such as wwwroot, Shared, and Pages. The Main method is shown as follows:

```
public class Program
{
    public static void Main(string[] args)
    {
        CreateHostBuilder(args).Build().Run();
    }

    public static IWebHostBuilder CreateHostBuilder(string[] args) =>
        BlazorWebAssemblyHost.CreateDefaultBuilder()
            .UseBlazorStartup<Startup>();
}
```

A Blazor application is made of components that each consist of a .cshtml Razor file. By default, the name of the .cshtml file is the name of the component. In the sample application, App is the name of the startup component whose source code is in app.cshtml, as shown here:

```
<Router AppAssembly=typeof(Program).Assembly />
```

At the minimum, the startup class indicates the App class to use.

```
public class Startup
{
    public void Configure(BlazorApplicationBuilder app)
    {
        app.AddComponent<App>("app");
    }
}
```

The App module just sets up the Blazor router component. In traditional Web sites (such as ASP.NET), routing takes place on the server, but in single-page applications it's preferably done via a dedicated client-side component. In Blazor, the router isn't made by JavaScript (as it is in, say, Angular), but, rather, is an instance of the internal Router class that parses all the classes in the referenced assembly, looking for those that implement the IComponent interface.

Hands-On Training for Developers

One Day, Two Day, and Virtual Training Options

Minneapolis | New York City⁺ Virtual Option
September 12 & 13, 2018 | October 1, 2018

Looking for training, but cannot sneak away from the office for a week-long conference? Look no further than Visual Studio Live! Hands-on Training (HOT). The experts at Visual Studio Live! are proud to announce two new HOT events - Minneapolis and New York City - with different offerings to best fit your training needs and accelerate your career.

REGISTER TODAY!

Add Visual Studio Live! HOT to Your 2018 Training Agenda if You Are Looking For:



Hands-On Training



Networking



Flexibility

[VSLIVE.COM/HOTMN](https://vslive.com/hotmn)

| [VSLIVE.COM/HOTNY](https://vslive.com/hotny)

The router is responsible for intercepting all navigation activity, both inward and outward, and making it happen. Clearly, Blazor components are expected to provide routing information for the router to invoke them appropriately. Blazor supports the same `@page` notation of Razor Pages in ASP.NET Core. In addition, it supports the `Route` attribute for component classes written in plain C#.

The Single Page

Usually, a single-page application starts from an `index.html` page. A Blazor application is no exception, and the single page is located under the `wwwroot` folder. The single page is expected to contain at least the following markup:

```
<app>
  <!-- initialization markup -->
</app>

<script src="_framework/blazor.
webassembly.js"></script>
```

The content of the `app` element is displayed during the initial loading of the Blazor platform, namely the first chunk of files in the list of **Figure 2**. You can have any valid HTML in the `app` element, most commonly a loading `.gif`. The script element is responsible for dynamically downloading the necessary `.dll` files.

The actual displayed content of the homepage is determined by the router and the remaining content of the `index.html` file found following the script element. Let's review the work done by the router.

The router checks the requested URL—the root of the deployed site—and looks for a registered component that can match the root URL. In the sample project, the `Pages` folder contains an `index.cshtml` file that begins with the following content:

```
@page "/"
<h1>Hello, Blazor!</h1>
Welcome to your first BLAZOR app.
```

The `@page` directive instructs the router about the template to pick up. If a `_ViewImports.cshtml` file is found in the `Pages` folder (and/or the root folder), its content is taken into account. In the sample project, the `_ViewImports` file in root folder contains a list of using statements, while the `_ViewImports`

Figure 3 C# Code of the DigitalClock Component

```
@functions {
    string currentTime = "N/A";
    string buttonAction = "N/A";
    string currentCss = "clock-notset";
    Timer timer;

    protected override async Task OnInitAsync()
    {
        InitTimer();
        StartTimer();
    }

    void startStop()
    {
        if (timer.Enabled)
        {
            StopTimer();
        }
        else
        {
            StartTimer();
        }
    }

    private Task TimerTick()
    {
        currentTime = DateTime.Now.ToLongTimeString();
        currentCss = "clock-working";
        this.StateHasChanged();

        return Task.CompletedTask;
    }

    private void InitTimer()
    {
        timer = new Timer(1000);
        timer.Elapsed += async (sender, e) => await TimerTick();
    }

    private void StartTimer()
    {
        buttonAction = "STOP";
        timer.Start();
    }

    private void StopTimer()
    {
        buttonAction = "START";
        timer.Stop();
    }
}
```

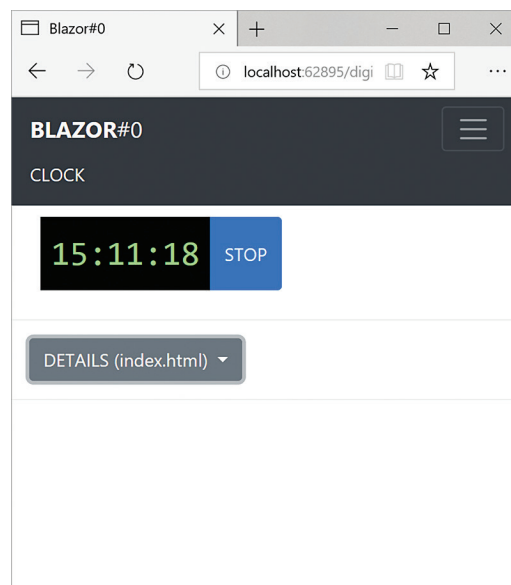


Figure 4 The Sample Application in Action

file in the `Pages` folder contains a reference to the layout template:

```
@Layout MainLayout
```

Subsequently, the content of the `MainLayout.cshtml` is processed and the `@Body` placeholder populated with the output of the selected Razor view (`index.cshtml`). In the sample project, the layout file (found in the `Shared` folder, as in ASP.NET Core) contains a typical Bootstrap 4 dark navbar element. Here's that code:

```
<nav class="navbar navbar-expand-lg
  navbar-dark bg-dark">
  ...
</nav>
<div class="container">
  @Body
</div>
```

Again, if you're coming from an ASP.NET MVC background, this programming pattern should look pretty familiar. In the navbar you typically have links. You can express links through the canonical `A` and `BUTTON` elements, or use one of the predefined (or custom) Razor tag helpers. In particular, Blazor comes with the `NavLink` helper component, which provides a friendlier syntax to express links that may or may not be plain URLs. In a single-page application, in fact, a link can be anything that the router can understand. Here's a fragment of Razor code used in the navbar of the sample application:

```
<div class="collapse navbar-collapse"
  id="navbarNav">
  <ul class="navbar-nav">
    <NavLink class="nav-link"
      href="digitalclock">
      DIGITAL CLOCK
    </NavLink>
  </ul>
</div>
```

The `href` attribute points to a URI that can just match the route to a component like "digitalclock," a separate CSHTML file that fully implements a digital clock component. The `digitalclock.cshtml` file is found under the `Pages` folder.

Building a Sample Component

A Blazor component is a self-contained chunk of HTML and C#. Any necessary CSS goes in separate files, reasonably named after the component itself. Note that Blazor doesn't do anything special with CSS and doesn't impose any convention. Dealing with CSS is

entirely up to you. Here's a digital clock implemented as a Bootstrap input group:

```
<div class="clock">
  <div class="input-group">
    <span class="@currentCss">@currentTime</span>
    <div class="input-group-append">
      <button class="btn btn-primary"
        type="button"
        onclick="@startStop">@buttonAction</button>
    </div>
  </div>
</div>
```

Razor expressions like `currentCss`, `currentTime` and `buttonAction` refer to component fields defined within the `@functions` section of the file. You can see these here:

```
@functions {
    string currentTime = "N/A";
    string buttonAction = "N/A";
    string currentCss = "clock-notset";
    Timer timer;
    ...
}
```

The `startStop` expression refers to a method being invoked when the Stop or Start button is clicked to start/stop the timer. **Figure 3** presents the full code of the `DigitalClock` component.

Figure 4 shows the program interface.

The demo presents a digital clock backed up by a .NET timer. Data binding is automatic for events triggered via an explicit user action (such as when a button is clicked), but not when the event originates programmatically, such as via a timer. In the latter case, you must call the method `StateHasChanged` to force the browser renderer to refresh the view. Here's the code of the timer tick handler within the digital clock Blazor component:

```
private Task TimerTick()
{
    currentTime = DateTime.Now.ToLongTimeString();
    currentCss = "clock-working";
    this.StateHasChanged();
    return Task.CompletedTask;
}
```

The full source code can be found at bit.ly/2LVCxA.

Wrapping Up

Blazor is an upcoming experimental framework for building .NET-based single-page applications running in the browser via WebAssembly. WebAssembly is a sandboxed way to deploy native binaries to a compliant browser—essentially all browsers shipped since mid-2017. Blazor uses Razor and C# to design views and includes a router and a composable user interface like most of the other popular JavaScript frameworks.

This article presents the foundation of a client-side application. In upcoming columns, I'll explore integration with the specific ASP.NET Core backend and Blazor extensibility points. You can read more about Blazor in Jonathan Miller's article that guides you through the quite necessary tasks for a client-side application of downloading JSON data from HTTP endpoints. ■

DINO ESPOSITO has authored more than 20 books and 1,000 articles in his 25-year career. Author of *"The Sabbatical Break,"* a theatrical-style show, Esposito is busy writing software for a greener world as the digital strategist at BaxEnergy. Follow him on Twitter: @despos.

THANKS to the following technical experts for reviewing this article:
Daniel Roth (Microsoft), Jonathan Miller

msdnmagazine.com



Automate your work using professional multi-platform Barcode SDK

VintaSoft Barcode .NET SDK

- .NET, WPF, WEB and Android SDK
- Generate and recognize 80+ 2D and 1D barcode types
- ISO quality tests, vector formats, many other advanced features



Professional SDK for building document management apps

- Image Viewer for .NET, WPF and WEB
- 100+ Image Processing and Document Cleanup commands
- PDF Reader, Writer, Visual Editor
- Manage interactive form fields and digital signatures in PDF
- Image Annotations
- JBIG2 and JPEG2000 codecs
- OCR and Document Recognition
- Forms Processing and OMR
- DICOM decoder
- TWAIN scanning

Free evaluation version
Royalty free licensing

www.vintasoft.com

VintaSoft® is a registered trademark
of VintaSoft Ltd.





TECHMENTOR

IN-DEPTH TRAINING FOR IT PROS

Training Conference for IT Pros

ROYAL PACIFIC RESORT AT
UNIVERSAL ORLANDO
December 2-7, 2018



Where IT Training Meets the Sunshine

TechMentor offers in-depth training for IT Pros, giving you the perfect balance of the tools you need today, while preparing you for tomorrow. Expect troubleshooting tips, performance optimization training, and best practices from peers and experts in the industry. Plus, there will be dedicated coverage of Windows PowerShell, core Windows Server functionality, Security, System Center and so much more. We'll see you in the sun!

TRACK TOPICS INCLUDE:

- Client and Endpoint Management
- PowerShell and DevOps
- Infrastructure
- Soft Skills for ITPros
- Security and Ethical Hacking
- Cloud (Public/Hybrid/Private)

**REGISTER
NOW**

**SAVE \$400 WITH
SUPER EARLY BIRD PRICING!
REGISTER BY OCTOBER 5**

Use Promo Code MSDN

A Part of Live! 360: The Ultimate Education Destination

6 CONFERENCES, 1 GREAT PRICE

Visual Studio  | SQL Server  | **TECHMENTOR** | Artificial Intelligence  | Office & SharePoint  | Modern Apps 



TECHMENTOREVENTS.COM/ORLANDO

EVENT PARTNERS



SILVER SPONSORS



EXHIBITORS



SUPPORTED BY



PRODUCED BY



AGENDA-AT-A-GLANCE

Client and Endpoint Management		PowerShell and DevOps	Infrastructure	Soft Skills for IT Pros	Security	Cloud (Public/Hybrid/Private)
START TIME	END TIME	TechMentor Full Day Hands-On Labs: Sunday, December 2, 2018				
7:30 AM	9:00 AM	Registration • Coffee and Morning Pastries				
9:00 AM	6:00 PM	TMS01 Hands-On Lab: Advanced Troubleshooting for IT/Ops Pros - Bruce Mackenzie-Low			TMS02 Hands-On Lab: Windows PowerShell Jump Start HOL - Jeffery Hicks	
2:00 PM	7:00 PM	Pre-Conference Registration - Royal Pacific Resort Conference Center				
START TIME	END TIME	TechMentor Pre-Conference Workshops: Monday, December 3, 2018				
7:00 AM	8:30 AM	Registration • Coffee and Morning Pastries				
8:30 AM	5:30 PM	TMM01 Workshop: Mastering the Sysinternals Toolkit - Sami Laiho			TMM02 Workshop: How to Successfully Manage a Fabric Using System Center 2016/2019? - Mikael Nystrom	
6:30 PM	8:00 PM	Dine-A-Round Dinner @ Universal CityWalk - 6:30pm - Meet at Conference Registration Desk to walk over with the group				
START TIME	END TIME	TechMentor Day 1: Tuesday, December 4, 2018				
7:00 AM	8:00 AM	Registration • Coffee and Morning Pastries				
8:00 AM	9:00 AM	TECHMENTOR KEYNOTE: OneDrive and M365 - Enabling Secure and Seamless Collaboration for a Untethered Workforce Stephen Rose, Sr. Product Manager, OneDrive for Business, Microsoft & Jason Moore, Head of OneDrive, Partner Group Program Manager, Microsoft				
9:15 AM	10:30 AM	TMT01 Everything You're Doing Wrong in PowerShell - Don Jones	TMT02 Shields Up! Managing the Windows Firewall with Advanced Security - Richard Hicks		TMT03 Image Factory: Automation Grand Deluxe - Mikael Nystrom	
10:30 AM	11:00 AM	Networking Break • Visit the EXPO - Pacifica 7				
11:00 AM	12:00 PM	LIVE! 360 KEYNOTE: To Be Announced - Pacifica 6				
12:00 PM	12:45 PM	Lunch • Visit the EXPO				
12:45 PM	1:30 PM	Dessert Break • Visit the EXPO				
1:30 PM	1:50 PM	TMT04 Fast Focus: Microsoft and Open Source - Jessica Deen		TMT05 Fast Focus: 5 Things Everyone Needs to Know About Azure - Peter De Tender		
2:00 PM	2:20 PM	TMT06 Fast Focus: Office 365 and Azure AD Security Quick Wins - Holly Lockhart & Oana Enache		TMT07 Fast Focus: AutoPilot Notes from the Field - Ami Casto		
2:20 PM	2:45 PM	Networking Break • Visit the EXPO - Pacifica 7				
2:45 PM	4:00 PM	TMT08 How I Would Audit Your Windows Security - Sami Laiho	TMT09 DevOps with Azure, Kubernetes, and Helm - Jessica Deen		TMT10 Consolidating DataCenters with Windows Failover Clusters - Bruce Mackenzie-Low	
4:15 PM	5:30 PM	TMT11 Securing Access to Office 365, SaaS and On-Premises Applications - Oana Enache & Holly Lockhart	TMT12 Using Desired State Configuration in Azure - Will Anderson		TMT13 Exploring Storage Replica in Windows Server 2019 - Mikael Nystrom	
5:30 PM	7:30 PM	Exhibitor Reception - Pacifica 7				
START TIME	END TIME	TechMentor Day 2: Wednesday, December 5, 2018				
7:30 AM	8:00 AM	Registration • Coffee and Morning Pastries				
8:00 AM	9:15 AM	TMW01 DevOps Making Snowflakes into Cattle - Don Jones	TMW02 No More Passwords! An Introduction to Windows Hello for Business - Richard Hicks		TMW03 Windows 10 Servicing with Traditional and Modern Options - Johan Arwidmark & Ami Casto	
9:30 AM	10:45 AM	TMW04 PowerShell: Under the Hood - Don Jones	TMW05 Techie Got Talent - Run Your Own Podcast / Webcast - Harjit Dhalwal & Nick Brattoli		TMW06 How to Setup and Maintain Config Manager in your Environment - Johan Arwidmark & Ami Casto	
10:45 AM	11:30 AM	Networking Break • Visit the EXPO - Pacifica 7				
11:30 AM	12:30 PM	LIVE! 360 KEYNOTE: Enterprise Transformation (And You Can Too) Donovan Brown, Principal DevOps Manager, Cloud Developer Advocacy Team, Microsoft				
12:30 PM	1:30 PM	Birds-of-a-Feather Lunch				
1:30 PM	2:00 PM	Dessert Break • Visit the EXPO				
2:00 PM	3:15 PM	TMW07 Conceptualizing Desired State Configuration Using Your Own Scripts - Will Anderson	TMW08 Securely Scripting PowerShell - Jeffery Hicks		TMW09 How to Use PowerShell to Become a Windows Management SuperHero - Petri Paavola	
3:15 PM	4:00 PM	Networking Break • Visit the EXPO • Expo Raffle @ 3:30 p.m. - Pacifica 7				
4:00 PM	5:15 PM	TMW10 Everything You Need to Know About Microsoft 365 - Harjit Dhalwal & Nick Brattoli	TMW11 The Zen of PowerShell Scripting - Jeffery Hicks		TMW12 Co-Management Status Quo - Panu Saukko	
7:30 PM	9:00 PM	Live! 360 Dessert Luau - Wantilan Pavilion				
START TIME	END TIME	TechMentor Day 3: Thursday, December 6, 2018				
7:30 AM	8:00 AM	Registration • Coffee and Morning Pastries				
8:00 AM	9:15 AM	TMH01 The PowerShell Core Tutorial - Don Jones	TMH02 What's New in Windows Server 2019 - Dave Kawula		TMH03 Securing Service Accounts the Modern Way - John O'Neil Sr.	
9:30 AM	10:45 AM	TMH04 Start Using JEA Today to Stop Overprivileged User Accounts - John O'Neil Sr.	TMH05 Deploying Application Whitelisting on Windows Pro or Enterprise - Sami Laiho		TMH06 Automated Troubleshooting Techniques in Enterprise Domains - Petri Paavola	
11:00 AM	12:00 PM	TECHMENTOR PANEL DISCUSSION: The Future of IT Sami Laiho and Dave Kawula (Moderators); Will Anderson, Johan Arwidmark, Ami Casto, and Don Jones				
12:00 PM	1:00 PM	Lunch on the Lanai - Lanai / Pacifica 7				
1:00 PM	2:15 PM	TMH07 Azure Security Unchained - Peter De Tender	TMH08 The Case of the Shrinking Data - Deduplication in Windows Server 2019 on ReFS - Dave Kawula		TMH09 Top 10 Configuration Manager Current Branch Mistakes (and How to Avoid Them) - Panu Saukko	
2:30 PM	3:45 PM	TMH10 Azure Stack, What You Need to Know - Peter De Tender	TMH11 The Weakest Link of Office 365 Security - Nestori Syynimaa		TMH12 How to Become a Community Rockstar - Learn How to Showcase Your Skills - Cristal Kawula	
4:00 PM	5:00 PM	Next? Live! 360 Networking Event Heidi Araya, Andrew Brust, Jeremy Clark, Ben Curry, Benjamin Day, Peter De Tender, Brent Edwards, Kevin Ford, Grant Fritchey, Esteban Garcia, Seth Juarez, Dave Kawula, Cristal Kawula, Sami Laiho, Thomas LaRock, Vishwas Lele, Rockford Lhotka, Karen Lopez, Matthew McDermott, Brian Randell, Paul Schaefflein, Jen Underwood, Rob Windsor				
START TIME	END TIME	TechMentor Post-Conference Workshops: Friday, December 7, 2018				
7:30 AM	8:00 AM	Registration • Coffee and Morning Pastries				
8:00 AM	05:00 PM	TMF01 Workshop: Notes from the Field on Storage Spaces Direct & Managing Windows Server with Project Honolulu (Windows Admin Center) - Dave Kawula		TMF02 Workshop: Hacking and Hardening Office 365 and Azure AD - Nestori Syynimaa		

Speakers and sessions subject to change



Time of the Season

Autumn is starting to turn here in Ipswich, Mass. As surely as the derivative of sine is cosine, the instant at which the darkness and light become momentarily equal is the instant at which they're changing most rapidly—half an hour in a week at its height, one waxing and the other waning.

The speed of this shortening photoperiod catches my attention, and I contemplate the cycles of the universe. I realize we modern-day humans are creatures of two cycles, wheels within wheels. Think of them as Ptolemaic astronomy described: The deferent, or outer cycle, of the year, imposed on us by celestial mechanics; combined with the epicycle, or inner cycle, of the week, solely a human creation. The last decade's technology has altered our adaptation to both. I'll discuss the latter today, and leave the former for another time.

Before on-demand streaming,
we consumed content when it
was broadcast. Each program
appeared at its customary phase
angle of the weekly epicycle,
affecting its content.

I wrote in my August 2017 column (msdn.com/magazine/mt493249) about the 50th anniversary of the classic The Beatles album, "Sergeant Pepper's Lonely Hearts Club Band." I noted that when it was released, the playback technology dictated that the order in which the artist placed the songs was generally the order in which the listener consumed them. Therefore, the artist had to compose this sequence carefully, and could create a synergistic whole if he did it well. I would argue that this careful composition is what elevates Sergeant Pepper into a masterpiece. Now that today's technology has decoupled that order, the artist no longer incurs this burden, but can no longer harness that synergy to improve the work.

This decoupling also affects broadcast media: movies, TV, radio. Before on-demand streaming, we consumed content when it was broadcast. Each program appeared at its customary phase angle of the weekly epicycle, affecting its content.

For example, when I was a kid, "Walt Disney's Wonderful World of Color" was shown at 7:30 on Sunday evenings. My siblings and I loved to watch it even on the black-and-white set we had at the

time. (That was so long ago that they showed Walt smoking a cigarette onscreen.) If we were all ready for bed (bathed, pajamas on, teeth brushed), we were allowed to watch Disney. But we knew that we had to go to bed when it was over, and our next conscious act would be waking up for school. The show signaled the end of the fun and the beginning of work. Its creators and producers had to, could, and did, tailor it to the particular mindset that its timeslot produced in its audience. Hence the term, "Disney Depression," now seldom understood by anyone under about age 50.

Each phase of the week had its own associated content. Saturday morning was for kids' cartoons. Saturday dinnertime gave us Garrison Keillor's "Prairie Home Companion" on PBS. Sunday morning was a huge wasteland of political talking head shows, barely relieved by the religious claymation show "Davey and Goliath" (whose characters now shill for Mountain Dew, see bit.ly/2LBD5LZ).

VCRs didn't change these relationships much because they were difficult to use. Even DVRs require forethought, a very rare commodity, then or now. But with on-demand video, every program that anyone can imagine queued up on sites legal and non, we can watch more or less anything at any time we want. It frees us as consumers, but removes a constraint that artists, at least the best ones, found helpful.

The artist had some notion about the particular stage of the weekly epicycle in which their creations would be consumed, so they could tailor them to the audience's mindset at that time—kids revving up on Saturday morning, kids bumming out on Sunday evening. They don't have this guidance anymore. They have to guess: Will their audience be at Friday happy hour, or on a Monday morning bummer? Will they be drinking champagne, or thinking about drinking Drano?

The advance of technology provides more choices for customers, which is generally reckoned a good thing. We now get to choose what to watch on Sunday nights. But as we enjoy these choices, proper contemplation requires that we think at least occasionally about the artistry that they now replace. And as Rolex still makes mechanical watches, there's still one program inseparably bound to its timeslot: We've just started another season of Monday Night Football. ■

Join me for my three-day workshop, "Building Joyful Xamarin Apps," at the Microsoft NERD center in Cambridge, Mass., Oct. 22-24. See joyfulxamarinapps.com for details.

DAVID S. PLATT teaches programming .NET at Harvard University Extension School and at companies all over the world. He's the author of 11 programming books, including "Why Software Sucks" (Addison-Wesley Professional, 2006) and "Introducing Microsoft .NET" (Microsoft Press, 2002). Microsoft named him a Software Legend in 2002. He wonders whether he should tape down two of his daughter's fingers so she learns how to count in octal. You can contact him at rollthunder.com.

**IS BAD DATA THREATENING
YOUR BUSINESS?**

CALL IN THE **FABULOUS 4**

IT'S CLOBBERIN' TIME...WITH DATA VERIFY TOOLS!

ADDRESS!

PHONE!

EMAIL!

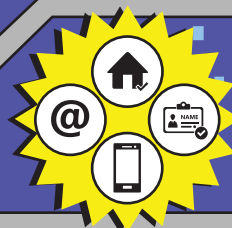
NAME!

Visit Melissa Developer Portal to quickly combine our APIs (address, phone, email and name verification) and enhance ecommerce and mobile apps to prevent bad data from entering your systems.

With our toolsets, you'll be a data hero – preventing fraud, reducing costs, improving data for analytics, and increasing business efficiency.

- Single Record & Batch Processing
- Scalable Pricing
- Flexible, Easy to Integrate Web APIs: REST, JSON & XML
- Other APIs available: Identity, IP, Property & Business

LET'S TEAM UP TO FIGHT BAD DATA TODAY!



Modern UI Made Easy



15% off on new purchases by August 31, 2018. See www.telerik.com/msdn for details.

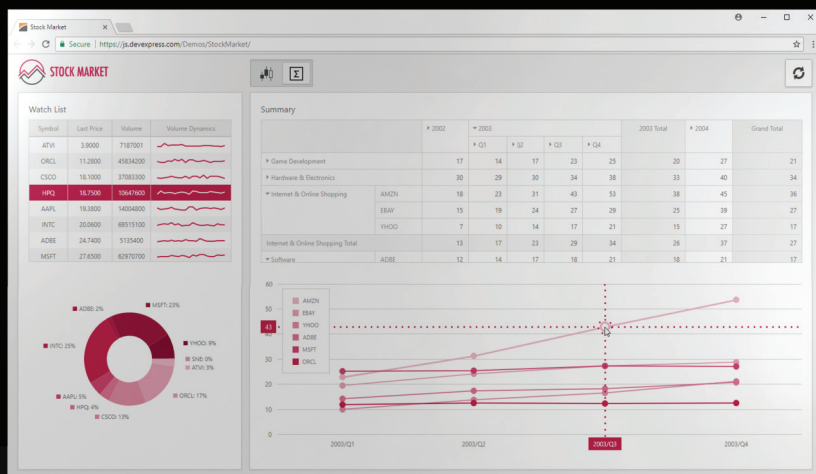
Building a modern UI for Web, Desktop and Mobile apps has never been easier
with our .NET, JavaScript & Productivity Tools

www.telerik.com/msdn



DXperience by DevExpress

Includes all DevExpress Web UI technologies
from ASP.NET & MVC to jQuery & Angular



USA
ComponentSource
650 Claremore Prof Way
Suite 100
Woodstock
GA 30188-5188
USA

Europe (UK)
ComponentSource
The White Building
33 King's Road
Reading, Berkshire
RG1 3AR
United Kingdom

Europe (Ireland)
ComponentSource
Unit 1E & 1F
Core B, Block 71
The Plaza, Park West
Dublin 12
Ireland

Asia-Pacific
ComponentSource
7F Kojimachi Ichihara Bldg
1-1-8 Hirakawa-cho, Chiyoda-ku
Tokyo
102-0093
Japan

www.componentsource.com

Sales Hotline:
(888) 850-9911

MSDN ISSUE



Your Next Great App Starts Here

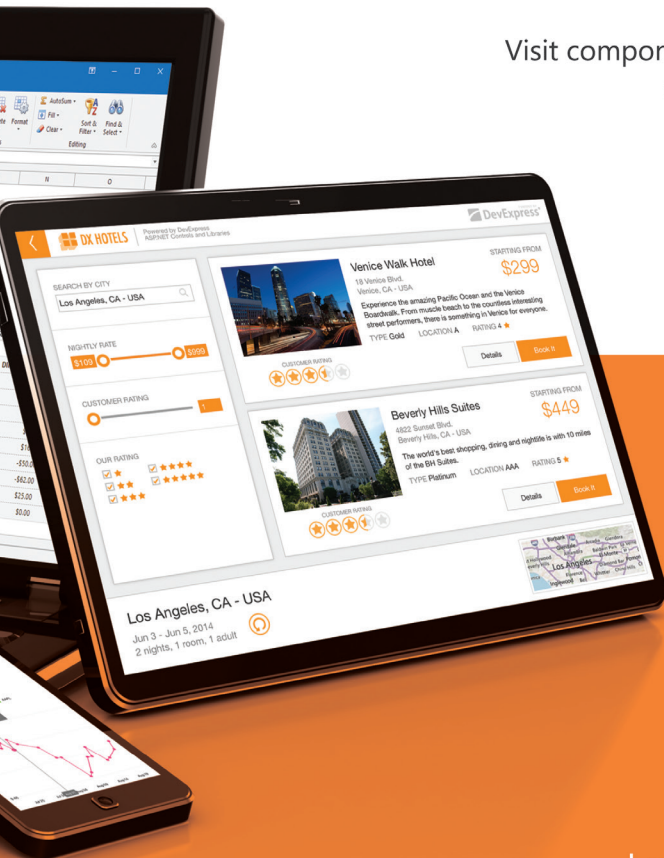
WIN ASP WPF   VCL



All trademarks or registered trademarks are property of their respective owners.
















Experience the DevExpress Difference Today














Visit componentsource.com/devexpress to see why DevExpress has been the #1 publisher on ComponentSource for **9 straight years**.



UI COMPONENTS / REPORTING / ANALYTICS
Learn more @ componentsource.com/devexpress

What's in this Issue?

/n software	 /n software Red Carpet Subscription	6
Aspose	 Aspose.Total for .NET	8
	 Aspose.Words for .NET	10
CData Software	 CData ADO.NET Driver Subscription	12
DevExpress	 DevExpress Universal	14
	 DevExtreme	16
GrapeCity	 ActiveReports	18
	 ComponentOne Studio	20
	 SpreadJS	22
Infragistics	 Infragistics Ultimate	24
	 Indigo.Design	26
	 Infragistics Professional	28
	 Infragistics Ignite UI for JavaScript	29
InstallAware	 InstallAware Studio Admin	30
	 InstallAware Studio	32

Syncfusion	 Syncfusion Essential Studio Enterprise	34
	 Syncfusion Essential Studio for Xamarin	36
Telerik	 Kendo UI	38
	 Telerik DevCraft Complete	40
	 Telerik DevCraft Ultimate	58
Text Control	 TX Text Control .NET for Windows Forms	42
	 TX Text Control .NET Server for ASP.NET	44
Actipro Software	 Actipro WPF Studio	46
Highsoft	 Highcharts Suite	48
IDM Computer Solutions	 UltraEdit	50
LEADTOOLS	 LEADTOOLS Document Imaging SDK	52
Nevron	 Nevron Vision for SSRS	54
SyncRO Soft	 oXygen XML Editor Enterprise	56

© 1996-2018 ComponentSource. All Rights Reserved. All prices correct at the time of press. Online prices may vary from those shown due to daily fluctuations and online discounts.



Official Supplier

As official and authorized distributors, we supply you with legitimate licenses directly from 300+ software publishers.



24/5 Customer Service

Need help to find the right software license, upgrade or renewal? Call, Email or Live Chat with our experts.



Trusted for Over 20 Years

Over 850,000 licenses delivered to Developers, SysAdmins, Corporations, Governments & Resellers, worldwide.

Open 24 hours a day, Monday to Friday

International Customer Service Centers in the US, UK, Ireland and Japan.

 United States	888 850 9911	 Finland	0800 1 18002	 Peru	0800 53288
 US Gov't Sales	888 850 9966	 France	0800 90 92 62	 Philippines	1800 1816 0315
 United Kingdom	0800 581111	 Germany	0800 186 07 06	 Poland	00800 442 1092
 Japan	0120 343 550	 Greece	00800 44121891	 Portugal	800 844 125
 Argentina	0800 666 0185	 Hong Kong	800 908 581	 Russia	810 800 2251 1044
 Australia	1 800 0 15292	 Hungary	06800 16674	 Singapore	800 810 2136
 Austria	0800 281 750	 India	000 800 44 01 455	 Slovenia	0800 80297
 Belgium	0800 7 5603	 Indonesia	001 803 00811 351	 South Africa	0800 99 1088
 Brazil	0800 891 6478	 Ireland	1 800 535 661	 Spain	900 93 8926
 Bulgaria	00800 115 4445	 Israel	180 924 2003	 Sweden	020 794 989
 Canada	(888) 850 9911	 Italy	800 790046	 Switzerland	0800 83 5305
 Chile	1230 020 6857	 Japan	0120 343 550	 Taiwan ROC	00 801 814313
 China (North)	10800 481 1661	 Korea, Rep. of	00798 14 800 6332	 Thailand	001 800 81 4 5257
 China (South)	10800 813 1594	 Malaysia	1 800 81 7261	 Turkey	00 800 4491 3617
 Colombia	01800 710 2043	 Mexico	01 800 681 1559	 United Kingdom	0800 581111
 Croatia	0800 222550	 Netherlands	0800 022 8832	 United States	888 850 9911
 Czech Republic	800 143 313	 New Zealand	800 449181	 Uruguay	000 401 902 38
 Denmark	80 88 17 20	 Norway	800 1 3685	 Venezuela	0 800 100 9103
 Ecuador	1 800 010 562	 Panama	00 1 800 203 1587	 Viet Nam	120 81 863

USA

ComponentSource
650 Claremore Professional Way
Suite 100
Woodstock
GA 30188-5188
USA

Tel: +1 770 250 6100
Fax: +1 770 250 6199

Europe (UK)

ComponentSource
The White Building
33 King's Road
Reading
Berkshire
RG1 3AR
United Kingdom

Tel: +44 118 958 1111
Fax: +44 118 958 9999

Europe (Ireland)

ComponentSource
Unit 1E & 1F
Core B, Block 71
The Plaza
Park West
Dublin 12
Ireland

Tel: +353 1 685 6704
Fax: +353 1 685 6725

Asia-Pacific

ComponentSource
7F Kojimachi Ichihara Bldg
1-1-8 Hirakawa-cho, Chiyoda-ku
Tokyo 102-0093
Japan

Tel: +81 3 3237 0281
Fax: +81 3 3237 0282

/n software Red Carpet Subscription

A comprehensive suite of Internet components. Any protocol, any platform, any IDE.



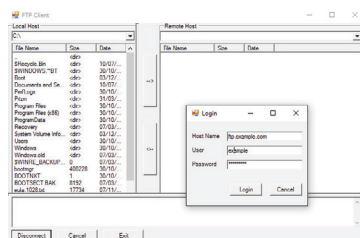
Publisher: /n software | Category: Communication & Messaging | ★★★★★

Components for every major protocol from FTP to IMAP to SNMP, SSL and SSH security, S/MIME encryption, Digital Certificates, Credit Card Processing, ZIP compression, Shipping and Tracking, and e-business (EDI) transactions.



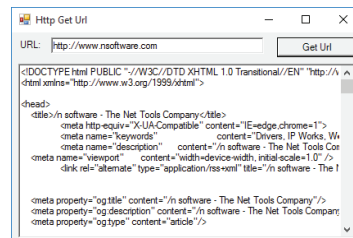
Encrypt/Decrypt Files and Emails

Easily implement strong encryption via major cryptographic standards. Features include encryption & decryption, secure message hashing, signing & signature verification and digital certificate generation.



Internet Communications

IP*Works! simplifies Internet development, providing easy-to-use, programmable components that facilitate tasks such as sending email, transferring files, managing networks, consuming Web Services, etc.



Exceptionally Responsive

Based on a highly optimized exceptionally responsive asynchronous socket architecture, the components consume a minimum of system resources while providing a high degree of compatibility with existing systems.



Prices from \$ 1,535.04

www.componentsource.com/nsoftware-red-carpet-subscription



We didn't invent the Internet...

...but our components help **you**
power the apps that bring it to business.

TOOLS • COMPONENTS • ENTERPRISE ADAPTERS

- **E-Business**
AS2, AS4, EDI/X12, OFTP ...
- **Credit Card Processing**
Authorize.Net, ACH, 3-D Secure ...
- **Shipping & Tracking**
FedEx, UPS, USPS ...
- **Accounting & Banking**
QuickBooks, OFX, SAP ...
- **Internet Business**
Amazon, PayPal, Google ...
- **Internet Protocols**
FTP, SMTP, IMAP, POP, WebDAV ...
- **Secure Connectivity**
SSH, SFTP, SSL, Certificates ...
- **Encryption & Certificates**
X.509, OpenPGP, SHA, S/MIME ...
- **Network Management**
SNMP, MIB, LDAP, Monitoring ...
- **Compression**
Zip, Gzip, Jar, AES, 7Zip ...

**2018 vol.1
UPDATE**

- New Products: API Integrator and IP*Works! IoT featuring MQTT
- Supports Windows IoT, Bluetooth Low Energy (BLE)
- New EDI Integrator Editions for C++, PHP, Node.js, and more

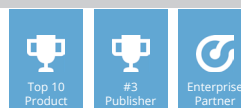
connectivity
powered by 

To learn more please visit → www.componentsource.com



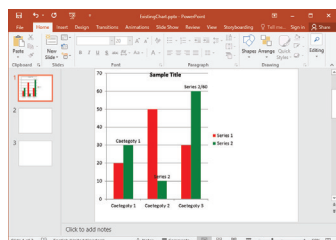
Aspose.Total for .NET

Open, create, convert, print & save files from within your own applications.



Publisher: Aspose | Category: Document & Text Processing | ★★★★★

Aspose.Total for .NET is a compilation of every .NET component offered by Aspose. It allows you to create a wide range of applications that work with file formats from Excel, Word, PowerPoint, Project, OneNote, Outlook, Visio and PDF.



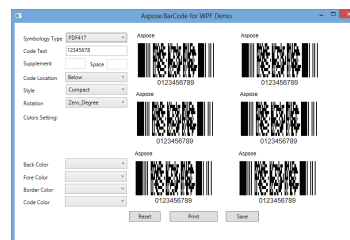
Aspose.Slides for .NET

Enables any .NET application to read, write, modify and convert PowerPoint documents, without the need for Microsoft PowerPoint. It can render presentation slides to PDF, XPS, HTML, Images and PDF Notes.

Product Name	Category Name	Quantity Per Unit	Units In Stock
Account Group	Category 1	10	10
Account Group	Category 2	10	10
Account Group	Category 3	10	10
Account Group	Category 4	10	10
Account Group	Category 5	10	10
Account Group	Category 6	10	10
Account Group	Category 7	10	10
Account Group	Category 8	10	10
Account Group	Category 9	10	10
Account Group	Category 10	10	10

Aspose.Cells for .NET

Enable .NET applications to create, modify, convert, render and print Excel spreadsheets without Microsoft Excel installed on the server. Supports XLS, XLSX, XLSM, XLTX/XLTM, HTML, CSV, ODS, PDF and more.



Aspose.BarCode for .NET

Allows developers to quickly and easily add robust and reliable barcode generation and recognition functionality to their Microsoft .NET applications. It supports WinForms and ASP.NET.



Prices from \$ 2,939.02
www.componentsource.com/aspose-total-net



File Format APIs

Open, Create, Convert, Print and Save files from your applications!



componentsource.com/aspose



Aspose.Words

Create, edit, convert or print Word documents (DOC, DOCX, RTF...)



Aspose.Cells

Create, Edit or Convert Excel worksheets (XLS, XLSX, ODS...)



Aspose.Pdf

Manipulate PDF file formats (PDF, PDF/A, XPS...)



Aspose.Slides

Create, edit or convert PowerPoint presentations (PPT, PPTX, ODP...)



Aspose.Email

Manipulate Email file formats (MSG, PST, EML...) and popular network protocols.



Aspose.BarCode

Generate or recognize barcodes (Code128, PDF417, Postnet...)

► [Aspose.Diagram](#) ► [Aspose.Note](#) ► [Aspose.3D](#) ► [Aspose.CAD](#) ► [Aspose.HTML](#) ► [Aspose.GIS](#)

USA: (888) 850 9911
United Kingdom: 0800 581111

France: 0800 90 92 62
Germany: 0800 186 07 06

Italy: 800 790046
Spain: 900 93 8926

sales@componentsource.com

Aspose.Words for .NET

Advanced .NET APIs to manipulate word processing documents.



Top 25
Product



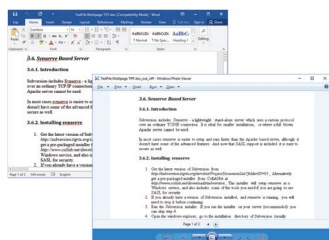
#3
Publisher



Enterprise
Partner

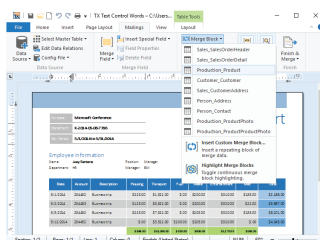
Publisher: Aspose | Category: Word Processing | ★★★★★

Perform a wide range of document processing tasks directly within your .NET applications, without using Microsoft Word. Supports DOC, OOXML, RTF, HTML, OpenDocument, PDF, XPS, EPUB and many other formats.



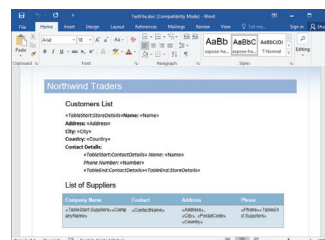
Render, Print & View Documents

Print or convert whole documents or even individual pages from Word documents to TIFF, PNG, BMP, JPEG, SVG or EMF images. You can also render (draw) any document page onto a .NET Graphic object.



Reporting and Mail Merge

Text Control combines the power of a programmable reporting tool and WYSIWYG word processor. Create mail merge and table reports, master-detail and sub-reports based on MS Word compatible report templates.



Advanced Document Processing

You can insert formatted text, images, tables and other content into a document. You can also combine documents, move sections between documents, and add images stored outside of the database.



Prices from \$ 979.02

www.componentsource.com/aspose-words-net



Manipulating Documents?

APIs to view, convert, annotate, compare, sign, assemble and search documents in your applications.

componentsource.com/groupdocs



 GROUPDOCS



GroupDocs.Viewer

View over 50 documents and image formats in any application using document viewer APIs.



GroupDocs.Annotation

Add annotations to specific words, phrases and any region of the document.



GroupDocs.Conversion

Fast batch document conversion APIs for any .NET, Java or Cloud application.



GroupDocs.Comparison

Compare two documents and get a difference summary report.



GroupDocs.Signature

Digitally sign Microsoft Word, Excel, PowerPoint and PDF documents.



GroupDocs.Assembly

Document automation APIs to create reports from templates and various data sources.

USA: (888) 850 9911
United Kingdom: 0800 581111

France: 0800 90 92 62
Germany: 0800 186 07 06

Italy: 800 790046
Spain: 900 93 8926

sales@componentsource.com

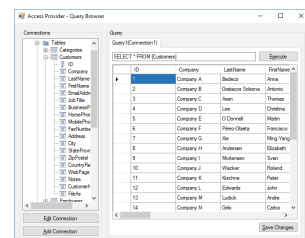
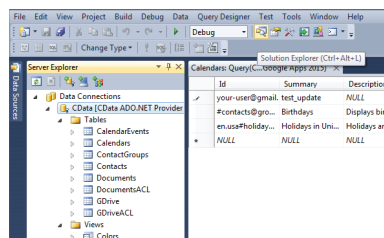
CData ADO.NET Subscription

Comprehensive access to data from ADO.NET.



Publisher: CData Software | Category: Data Access

Real-time data integration from 80+ SaaS, NoSQL, and Big Data sources. Includes a single SQL interface to data that insulates users from the challenges and complexities of integrating with individual APIs, SDKs, and services.



Connect .NET Apps to Data

Comprehensive access to application, database, and Web API data through easy-to-use tools. Whether you are building Desktop, Web, or Mobile apps you can use the drivers for fast and secure access to data.

Visual Studio Integration

The ADO.NET Data Providers can be used to access and explore data directly from the Visual Studio Server Explorer. These constructs return live data that you can work with directly from within Visual Studio.

Access Data from any .NET App

Use objects to connect to data just as you would any traditional database, through the Visual Studio Server Explorer, in code, and in data controls like DataGridView, GridView and DataSet.



Prices from \$ 979.02
www.componentsource.com/cdata-ado-net-subscription



Know SQL?

Connect to 80+ data sources (NoSQL, Big Data, SaaS, etc.) through standard SQL queries.



ODBC



JDBC



ADO.NET



SSIS



CLOUD



BIZTALK

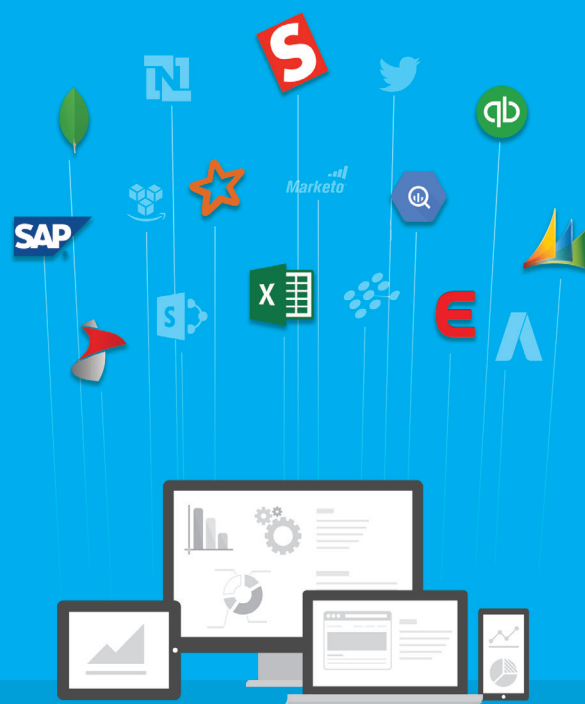


EXCEL



MOBILE

SQL is the language of data, and our state-of-the-art Drivers let you leverage the full power of SQL to connect with hundreds of applications, databases, and APIs. Through standards-compliant Driver technologies like ODBC, JDBC, ADO.NET, & OData, you can read, write, and update live data, from any data source, using standard SQL queries.



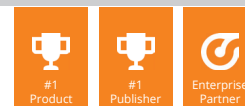
Learn more: www.componentsource.com/cdata



Copyright © 2018 CData Software, Inc. All rights reserved. All trademarks and registered trademarks are the property of their respective owners.

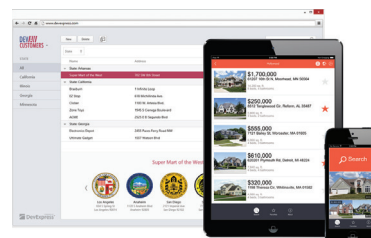
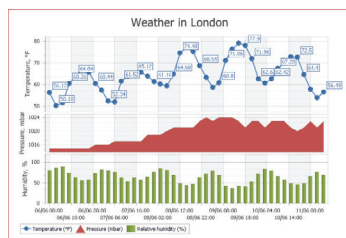
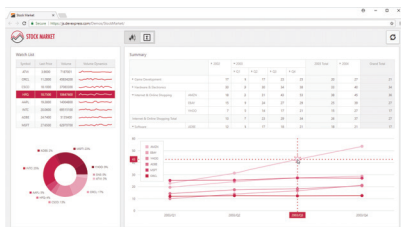
DevExpress Universal

All DevExpress Visual Studio products in one integrated subscription.



Publisher: DevExpress | Category: Presentation Layer | ★★★★★

Includes charts, data grids, spreadsheets, calendars, schedulers, diagrams, navigation, text editors, PDF, maps, gauges, tiles and data editors. Supports ASP.NET WebForms & MVC, WinForms, WPF, Windows 10 and HTML5/JS.



Web UI Controls

Deliver best-of-breed user experiences and emulate the UI of today's most popular web based business productivity apps with DevExpress components for ASP.NET, MVC, Angular, and React.

Comprehensive 2D & 3D Charting

Optimized for both performance and flexibility, DevExpress charts are highly customizable controls which can be leveraged in many business scenarios, from real-time charting to multi-dimensional data analysis.

HTML5/JS Framework

A JavaScript development framework that allows you to create store-ready, multi-device applications across platforms and devices. Each widget includes a jQuery plugin and bindings to AngularJS and KnockoutJS.

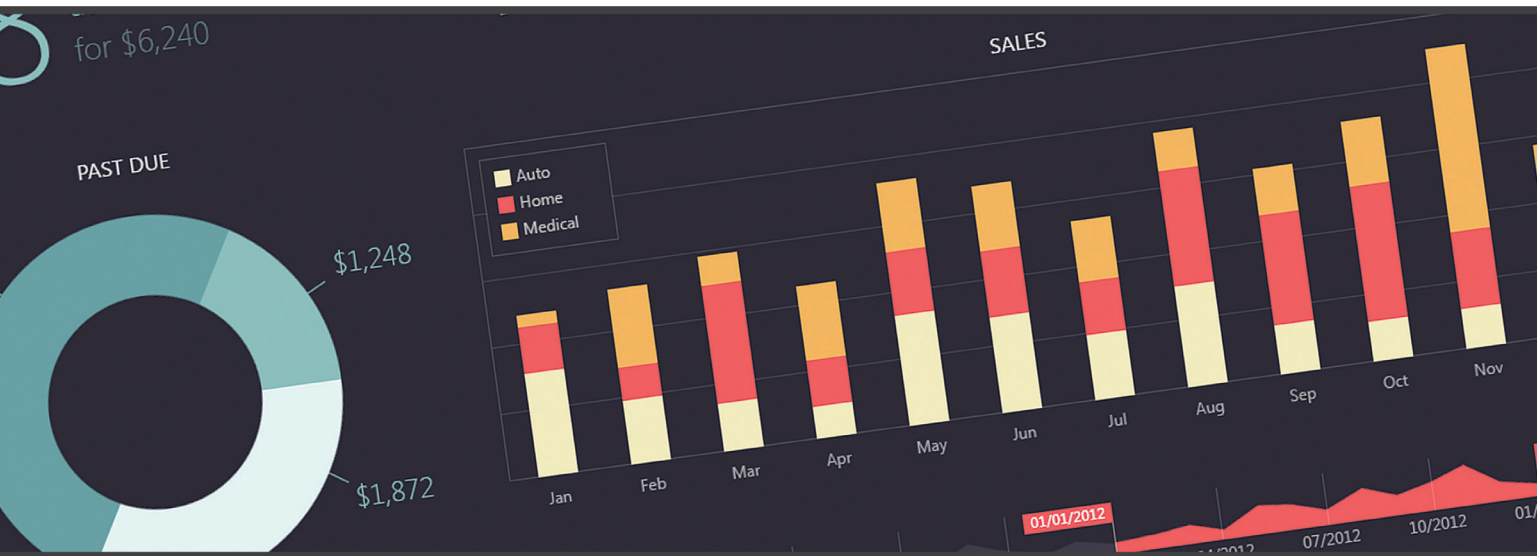


Prices from \$ 2,111.99
www.componentsource.com/devexpress-universal



Dashboards & Analytics

DevExpress Universal ships with everything you'll need to create information-rich decision support systems and to distribute your dashboard solutions royalty-free.



Your Next Great Dashboard Starts Here

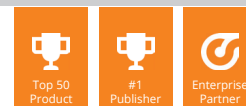
Learn how you can create fully customizable Dashboards with our flexible data visualization tools.
componentsource.com/devexpress-universal



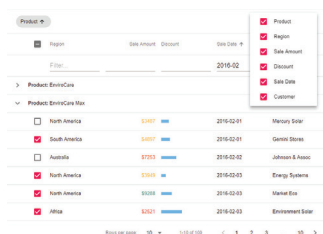
DevExtreme

High-performance HTML5/JavaScript Widgets for next-generation app development.

Publisher: DevExpress | Category: Presentation Layer | ★★★★★

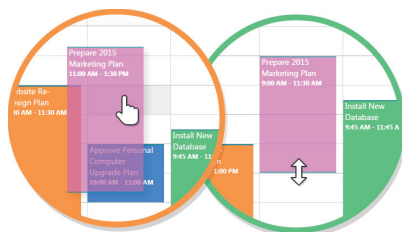


Use your familiarity with JavaScript and the Visual Studio IDE to build store ready apps that run across iOS, Android, and Windows Phone. Includes grids, charts, treelist, maps, forms, scheduler and more.



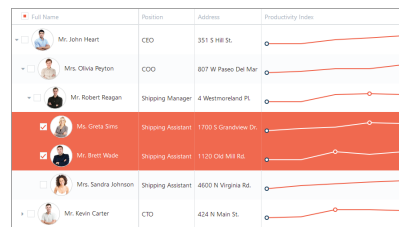
Grids, Trees and Editors

The blazing-fast DevExtreme Data Grid is a feature-rich data shaping and editing client-side widget which allows your end users to easily manage information and display it on-screen as business requirements dictate.



Scheduling and Navigation

The DevExtreme Scheduling widget ships with four pre-built views (Day, Week, Work Week and Month). Your users can quickly switch between each view via its integrated calendar view selector.



Tree List

The DevExtreme Tree List is an intuitive and easy to use widget that combines the power of a traditional HTML5 Grid and a TreeView in a single UI element.



Prices from \$ 479.99
www.componentsource.com/devextreme



Reporting Made Easy

Inform and analyze with absolute ease. DevExpress Reporting includes an enterprise-ready reporting platform with integrated Windows & Web report designers.

INVOICE

NORTHWIND TRADERS

One Portland Way, Twin Pointe, WI 53156 Phone: 1-206-555-1417 Fax: 1-206-555-5938

Date: {0:d}

Ship To: [Invoices.ShipName]
[Invoices.ShipAddress]
ShipCityRegionPostalCode
[Invoices.ShipCountry]

Bill To: [Invoices.Customer.CompanyName]
[Invoices.Address]
CityRegionPostalCode
[Invoices.Country]

Order ID:	Customer ID:	Salesperson:	Order Date:	Required Date:	Shipped Date:	Ship Via:
[Invoices.OrderID]	[Invoices.CustomerID]	[Invoices.SalespersonID]	[Invoices.OrderDate]	[Invoices.RequiredDate]	[Invoices.ShipDate]	[Invoices.ShipperID]

Quantity:	Unit Price:	Discount:	Extended Price:
[Invoices.Quantity]	[Invoices.UnitPrice]	[Invoices.Discount]	[Invoices.ExtendedPrice]

Text

- DATA BINDING
- SUMMARY

Angle: 0

Bookmark

Parent Bookmark

Select...

Auto Width

Can Grow ☒

Can Shrink ☐

Multiline ☒



Your Next Great Business Report Starts Here

See the power and flexibility of our reporting platform in action.

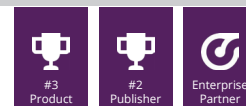
componentsource.com/devexpress-reporting



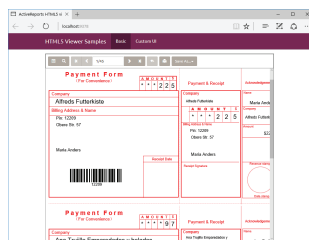
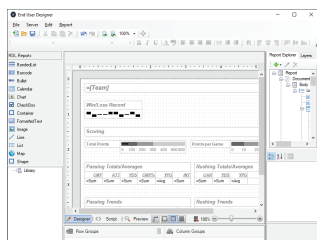
ActiveReports

A flexible and extensible platform for every kind of reporting.

Publisher: GrapeCity | Category: Reporting | ★★★★★



Design, customize, publish, and view data in your business applications. Create everything from simple invoices to complex statistical data visualizations, and you can even enable your end users to design reports for themselves.



Customizable End-User Designer

Embed a report design environment in your own WinForms applications, giving end users the power to design their own reports in a code-free ad-hoc reporting environment.

Report Types for Every Need

ActiveReports helps developers design accounting, analytical, budget, risk analysis, scientific, calendar, scorecard, insurance, healthcare, clinical, stock analysis, and other business reports for their apps.

Multi-Platform Report Viewers

Embed the fully customizable viewers in your HTML5, WPF, WinForms or ASP.NET MVC applications. PDF, Excel, Word and other export formats support opening report output in other programs.



Prices from \$ 1,575.02
www.componentsource.com/activerports-developer



GrapeCity | Developer Solutions



ActiveReports

.NET Reporting Platform for Any Business Need



Create interactive
ad-hoc reports



Distribute reports to
all devices, **royalty-free**



Customize with the
extensive .NET API



Integrate **charts** with
data visualization controls



Schedule and manage
enterprise reporting
with the **reports server**



Design and publish server
reports with the **web-based**
designers and portals

© 2018 GrapeCity, Inc. All rights reserved. All other product and brand names are trademarks and/or registered trademarks of their respective holders.

Learn more at componentsource.com/activereports

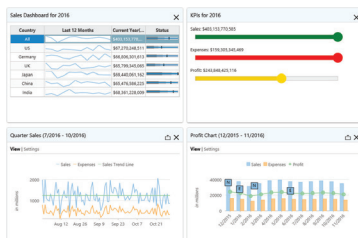
ComponentOne Studio Enterprise

A complete collection of award-winning .NET UI controls for mobile, web and desktop.

Publisher: GrapeCity | Category: Presentation Layer | ★★★★★

Includes grids, data management, data visualization, reporting, documents, scheduling, input, editing, navigation, layout and utilities. Supports WinForms, WPF, UWP, ASP.NET MVC, ActiveX, LightSwitch, Silverlight, ASP.NET Web Forms.

Line	Order	Price	Weight	Cost	Material	Rating	Download
Line 123	Line 123						
Line 504	Stone	Yellow	\$430	\$240	\$120.00	1219	2.00
Line 505	Stone	Yellow	\$430	\$240	\$120.00	1219	2.00
Line 517	Stone	Yellow	\$100	\$60	\$20.00	2818	4.00
Line 520	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 523	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 524	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 525	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 526	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 527	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 528	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 529	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 530	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 531	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 532	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 533	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 534	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 535	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 536	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 537	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 538	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 539	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 540	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 541	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 542	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 543	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 544	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 545	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 546	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 547	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 548	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 549	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 550	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 551	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 552	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 553	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 554	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 555	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 556	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 557	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 558	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 559	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 560	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 561	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 562	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 563	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 564	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 565	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 566	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 567	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 568	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 569	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 570	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 571	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 572	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 573	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00
Line 574	Stone	Yellow	\$120	\$70	\$24.00	1334	4.00



Industry-Leading FlexGrid

Sort, group, filter, merge cells, freeze columns, and import or export to Excel with FlexGrid, an industry-leading flexible data grid. Get power, flexibility, and speed in a single lightweight grid control.

Powerful Data Visualization

Whether you need 60+ charts, Gantt views, pivot tables, gauges, maps, or sparklines, ComponentOne Studio's data visualization controls handle large data sets and impress your users with professional design.

Engaging User Interactions

The UI and navigational control set offers intuitive displays of information with several forms of interaction. Edit, sort, delete, build menus and toolbars, and easily apply themes when crafting tailored UIs.



Prices from \$ 1,472.58
www.componentsource.com/componentone-studio





ComponentOne

Deliver brilliant UI in less time with this complete collection of .NET UI controls



Select from 300+ extensible controls with easy-to-use universal API



Improve your apps with the industry's best grids, charts, and reports



Reduce app bloat with the small footprint



Take advantage of the global support and community

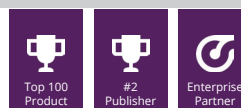


Distribute apps royalty-free

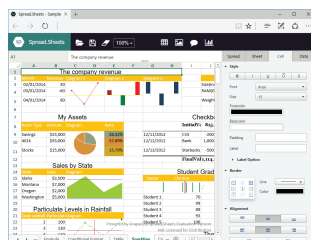
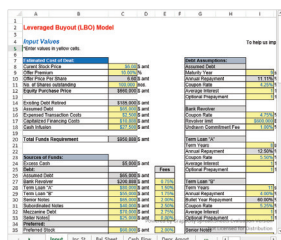
SpreadJS

High-speed Excel-like JavaScript spreadsheet components for enterprise apps.

Publisher: GrapeCity | Category: Spreadsheet



Deliver full-featured Excel-like spreadsheet, input forms, numerical models, and finance dashboards in less time with SpreadJS' dependency-free spreadsheet and data presentation components.



Formula	Description
<code>=BETA.DIST(2,8,10,TRUE,1,3)</code>	Cumulative beta probability density function
<code>=BETA.DIST(2,8,10,FALSE,1,3)</code>	Beta probability density function
<code>=CEILING.PRECISE(4.3)</code>	Rounds 4.3 up to the nearest multiple of 1.
<code>=CEILING.PRECISE(-4.3)</code>	Rounds -4.3 up to the nearest multiple of 1, because the number is negative.
<code>=CEILING.PRECISE(4.3, 2)</code>	Rounds 4.3 up to the nearest multiple of 2.
<code>=CHISQ.DIST(0.5,1,TRUE)</code>	The chi-squared distribution for 0.5, returns distribution function, using 1 degree of freedom.
<code>=CHISQ.DIST(2,3,FALSE)</code>	The chi-squared distribution for 2, returns density function, using 3 degrees of freedom.
<code>=CHISQ.INV(0.95,1)</code>	Inverse of the left-tailed probability of the chi-squared distribution, using 1 degree of freedom.
<code>=CHISQ.INV(0.6,2)</code>	Inverse of the left-tailed probability of the chi-squared distribution, using 2 degrees of freedom.
<code>=CONFIDENCE.TP(0.05,1,50)</code>	Confidence interval for the mean of a population, sample size of 50, with a 5% significance level deviation of 1. This is based on a Student's t-distribution.

Flexible JavaScript Spreadsheet

Deliver a familiar, fast, Excel-like experience. Import and export native Excel files with features like formula engine, sorting, filtering, input controls, tables, charts, and sparklines. SpreadJS includes full support for Angular, React, and Vue.

Data Display and Visualization

With full Excel chart support, sparklines, and a free-form grid layout, you'll build extensive data visualization dashboards that include performance indicators, data bars, color scales, and conditional formatting.

Function and Calculation Engine

Choose from over 460 built-in functions and apply them to individual cells, rows, columns, or entire sheets. With the Excel-like formula text box and your own custom formulas, you'll add power and speed to your apps.



Prices from \$ 1,476.52
www.componentsource.com/spreadjs





Spread

Deliver high-performance, high-power spreadsheet apps for JavaScript



Design **spreadsheets, charts, and input forms** code-free with **desktop designer apps**



Generate **Excel documents** with the **fastest spreadsheet API** available



Create **Excel-like** interactive data views and forms



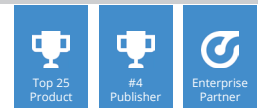
Build apps for **risk analysis, financial models, budgeting, and business performance**



Work with **462 Excel functions**, the most available

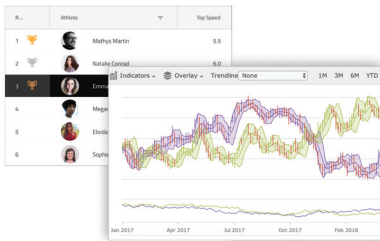
Infragistics Ultimate

UX design & enterprise app development for Web, desktop and mobile.



Publisher: Infragistics | Category: Presentation Layer

Includes 100+ UI controls & components, plus Grids and Charts on WPF, Windows Forms, ASP.NET MVC, jQuery & Angular. New in 2018 is Indigo.Design, a unified platform for visual design, UX prototyping, and code generation.



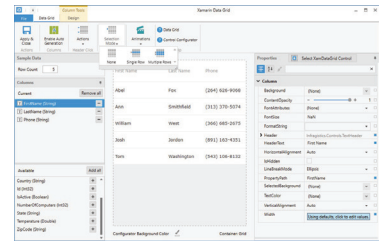
Fast Grids and Charts

Scroll, Zoom & Pan millions of data points on demand or in real-time with millisecond response time in WPF & Angular.



Best-of-Breed UI

No-compromise needed - Infragistics Ultimate delivers the most important controls for your most demanding apps on all modern platforms.



Advanced CLI Tools

With RAD visual configurators for mobile & WPF, command-line tools (CLI) for Angular & JavaScript, you write less code and get more done with Ultimate.



Prices from \$ 1,955.10
www.componentsource.com/infragistics-ultimate

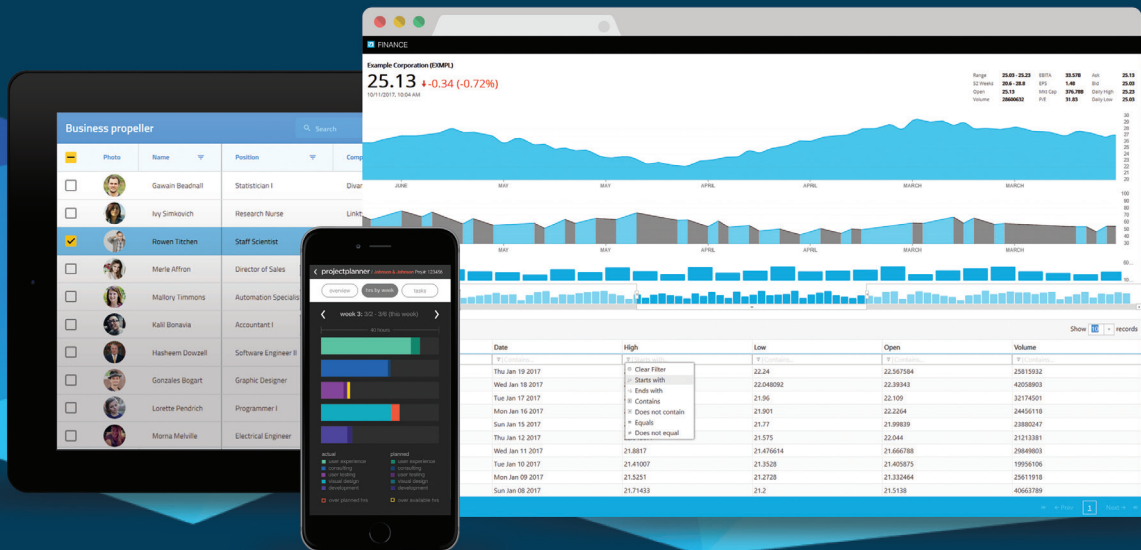




Faster Paths to Amazing Experiences

Infragistics Ultimate includes 100+ beautifully styled, high performance grids, charts & other UI controls, plus productivity tools for building web, desktop and mobile apps.

All new unified platform for UX, prototyping, usability testing, and code generation.



Angular | JavaScript / HTML5 | ASP.NET | Windows Forms | WPF | Xamarin

Learn more: componentsource.com/infragistics

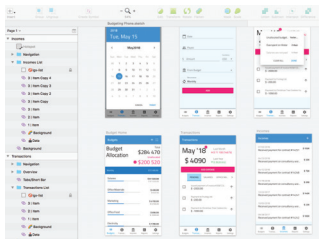
Indigo.Design

Unparalleled productivity for teams. Pixel-perfect Angular code from Sketch designs.



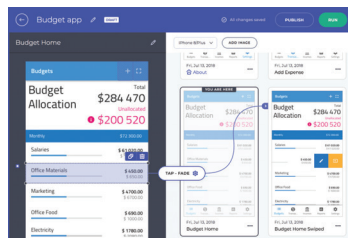
Publisher: Infragistics | Category: Software Architecture & Design

Designer meets developer with Indigo.Design a unified platform for visual design, UX prototyping, code generation, and app development.



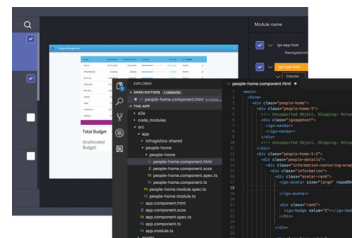
Design to Match Your Brand

Create best-in-class UI designs using the expressive Indigo Design System with Sketch UI Kits. With 50+ components, 30+ UI patterns and complete app scenarios, getting started is a breeze.



Share, Collaborate & Test

Import Sketch files into the cloud and share user flows and interactions on any device. Record & playback usability studies and get real-time reporting & analytics to see how users interact with your designs.



Get Runnable Code with 1-Click

Everything you craft in Sketch from the design system matches the Ignite UI for Angular components. With the click of a button, generate high-quality HTML, CSS, and Angular code from your design.

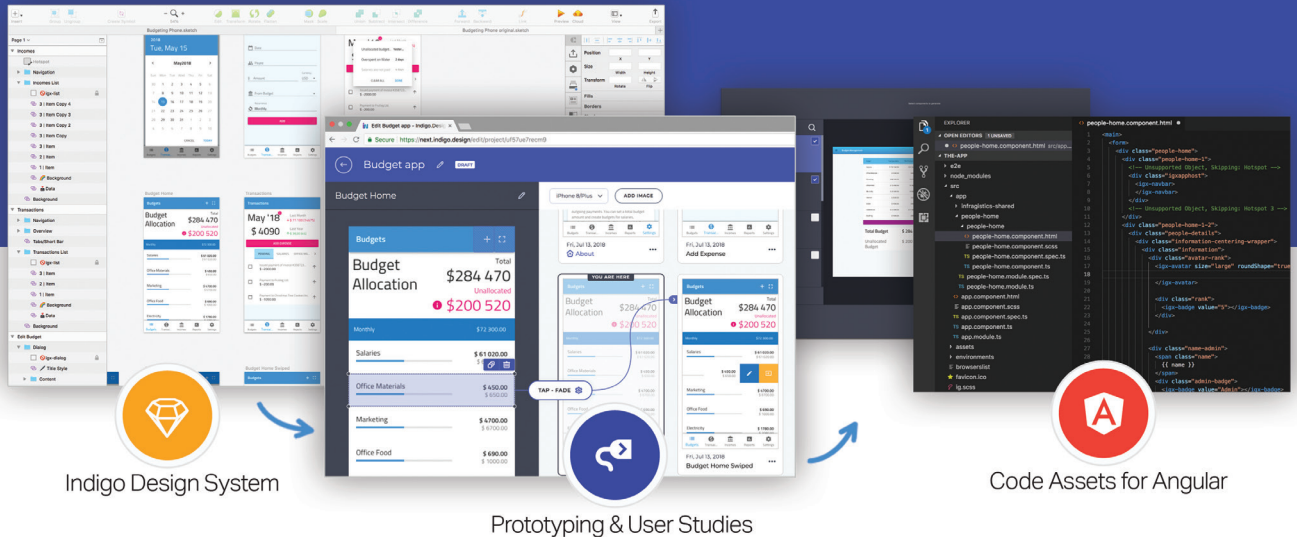


Prices from \$ 975.10
www.componentsource.com/indigo-design



Indigo.Design

A Unified Platform for Visual Design, UX Prototyping, Code Generation & App Development

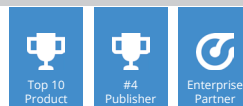


The world's first, and only, end-to-end comprehensive design to code platform for UX Designers, Visual Designers & Developers that generates pixel-perfect Angular apps from Sketch designs.

Learn more: componentsource.com/infragistics

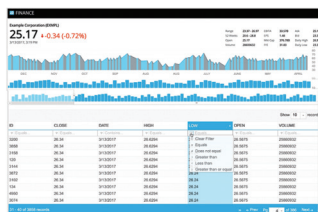
Infragistics Professional

Build functional and stylish Enterprise-ready applications.

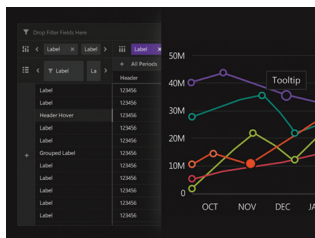


Publisher: Infragistics | Category: Presentation Layer | ★★★★★

Get performance, ease of use, and styling with a powerful suite of UI components that allows you to develop across all browsers, devices, and platforms. Contains hundreds of UI controls, including grids, charts, maps and MS Office-compatible frameworks.



AngularJS AngularJS KnockoutJS React Bootstrap



JavaScript/HTML5 & ASP.NET MVC

Build high-performing, responsive Web applications in JavaScript/HTML5 for Angular and React frameworks. Get a jump-start on the most demanding Web applications with ASP.NET MVC server-side wrappers.

Touch-friendly WPF Controls

Bring modern, engaging apps for desktop and touch-screen devices to market while shortening development time. Includes fast data grids, charts, dynamic data visualization, scheduling, styling and themes.

Modern Web Development Tools

With full support for Angular and Bootstrap, Ignite UI is the complete HTML & JavaScript toolkit that helps you build modern browser experiences on the desktop, tablet or phone.



Prices from \$ 1,465.10
www.componentsource.com/infragistics-professional



Create high-performance, touch-first, responsive apps.

A complete HTML5 & JavaScript toolkit to build modern browser experiences on any device - desktop, tablet or phone. Create powerful Angular and React Web applications with fast, easy-to-use UI components and productivity tools.



Prices from \$ 881.02
www.componentsource.com/infragistics-ignite-ui



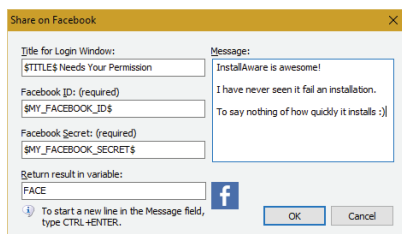
InstallAware Studio Admin

A powerful Windows software installation solution with Web-based automation support.

Publisher: InstallAware | Category: Release Automation & Management

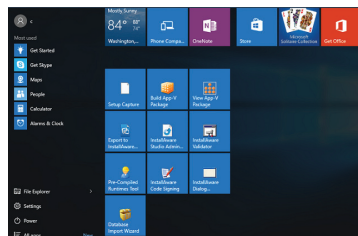


Provides the advantages of true Windows Installer rapid development productivity, plus repackaging support and a Web based Automation Interface.



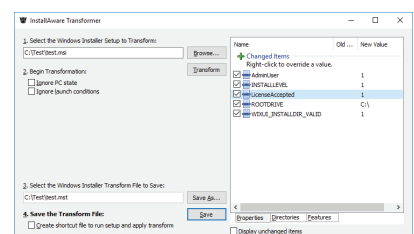
Post to Social Networks

Query for social network credentials at any time while your setup is running on end-user systems. Tweet on Twitter and Share on Facebook, before continuing with your installation.



Latest Technology Support

InstallAware supports Windows 10 and Visual Studio 2017. Display installation progress on the Windows Taskbar, and pin apps to the new Windows 10 Start Menu Live Tiles.



InstallAware Transformer

Give non-developers a point-and-click way to create MST transform files for any MSI setup. Any user may run through the setup, and instead of installing the package, InstallAware creates an MST transform file.



Prices from \$ 3,919.02
www.componentsource.com/installaware-studio-admin



NEW!

InstallAware X8

InstallAware X8 is the first and only installer with Microsoft Fluent Design System support, sporting an Acrylic Materials theme. InstallAware X8 is also the first and only installer with a 1.5 GB compression dictionary, for effective deduplication.



ACRYLIC MATERIALS (WITH FALLBACK TO AERO GLASS)

The best looking installations on all platforms, including downlevel.



UNLIMITED DIALOG EDITOR UNDO/REDO

Unleash your creativity without consequences for the safest design experience.



1.5 GB COMPRESSION DICTIONARY

The smallest setups, saving both you and your clients bandwidth.



INSTANT EXTRACTION

Even multi-gigabyte setups launch instantly, with just-in-time payload extraction.



VISUAL STUDIO TEAM SERVICES/TEAM FOUNDATION SERVER

Get projects from servers instantly without doing any folder mapping.

InstallAware is the most flexible platform for traditional and agile development teams creating Windows and Azure software installers, as well as APPX Universal Windows Platform, Microsoft App-V Virtualization, and agentless/royalty-free InstallAware Virtualization packages.

InstallAware additionally supports the upcoming new MSIX packaging technology, in close partnership with Microsoft.



www.componentsource.com/installaware

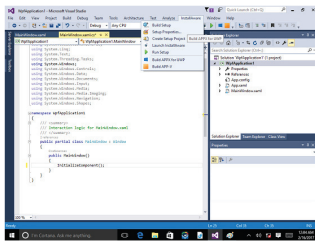
InstallAware Studio

A powerful software installation solution for deploying products, patches and updates.



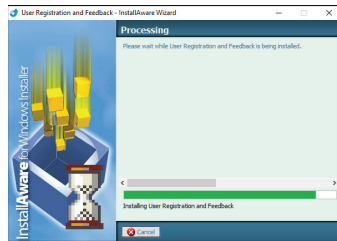
Publisher: InstallAware | Category: Release Automation & Management

Supports the latest technologies and provides the advantages of true Windows Installer rapid development productivity. Enables MSIcode scripting for rapid setup development without the steep learning curve of other setup solutions.



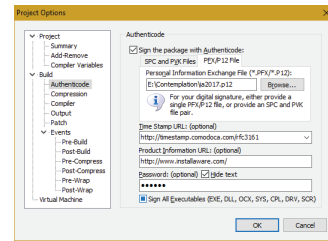
One-Click APPX Builds from the Visual Studio Toolbar

Click once in the InstallAware Toolbar's APPX Builder to take your active Visual Studio project, and build an APPX package out of it, ready for submission to the Windows Store.



Faster, More Reliable Installs

Ships with a Native Code Setup Engine, a faster and more reliable alternative to Windows Installer. The Native Engine with InstantInstall Acceleration creates Windows Installers faster by an order of magnitude.



SHA 256 Code Signing

InstallAware applies a smart algorithm to determine whether files are going to be singly signed, doubly signed, or skipped. All signing happens automatically to comply with Microsoft's code signing standards.



Prices from \$ 1,959.02
www.componentsource.com/installaware-studio







About InstallAware

InstallAware Software, founded in 2003, is the leading Cloud Infrastructure Company with its laser sharp focus on bullet-proof enterprise software deployment. InstallAware has received multiple awards from Microsoft, ComponentSource, WindowsITPro, SDTimes (Leader of the Software Development Industry), Visual Studio Magazine (Reader's Choice), among others.

InstallAware X8 is available in a complimentary edition for all Visual Studio users and paid editions. For more information visit: <https://www.componentsource.com/installaware>

-  **INSTALLTAILOR MST TRANSFORM CREATION**
-  **ONE-CLICK BINARY DIFFERENTIAL PATCHING**
-  **MICROSOFT VISUAL STUDIO INTEGRATION**
-  **AUTOMATED VIRTUAL MACHINE TESTING**
-  **PROGRAMMATIC APPLICATION PINNING**



www.componentsource.com/installaware

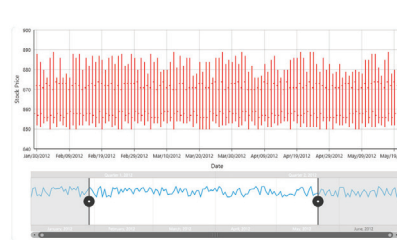
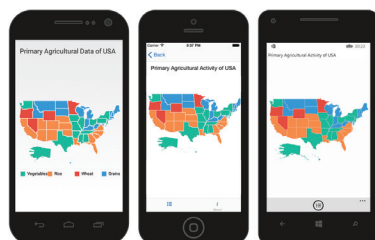
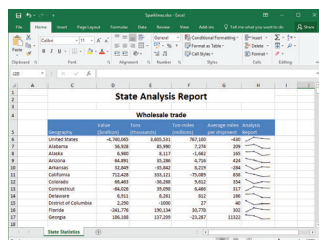
Syncfusion Essential Studio Enterprise

Deliver better solutions with a complete suite of .NET & JavaScript controls.



Publisher: Syncfusion | Category: Presentation Layer | ★★★★★

Includes 800+ components across Windows Forms, WPF, ASP.NET MVC, ASP.NET Web Forms, LightSwitch, Silverlight, iOS, Android, Windows Phone, Xamarin, JavaScript and UWP. No royalties, run-time, or server deployment fees.



Read and Write Excel Files

Enable any .NET application to create, read, write and process Microsoft Excel files. Features access to each aspect of the file, blazing performance, no server deployment fees and easy migration from Office Automation.

Xamarin Controls

A collection of enterprise-grade Xamarin.Forms components for building Mobile applications. It includes all the UI controls that are typically required for building line-of-business applications.

High Performance Charting

Plot a wide range of chart types from line charts to specialized financial charts. Syncfusion's rich feature set includes data binding, multiple axes, trackball, drill-down operations and zooming.



Prices from \$ 1,795.50
www.componentsource.com/syncfusion-essential-studio-enterprise





CALLING ALL C# DEVELOPERS!



WHY SYNCFUSION?

- Built from scratch using Xamarin
- Optimized for performance
- Frequent updates – 4 per year

Meet the challenge of cross-platform development

- ✓ 100+ Xamarin controls, including the fastest chart and grid available.
- ✓ Build apps that run across iOS, Android, and Windows using a single C# codebase.
- ✓ Seamless integration with Visual Studio.

GET STARTED TODAY!

componentsource.com/syncfusion-essential-studio-for-xamarin



Syncfusion Essential Studio for Xamarin

Comprehensive component suite for Xamarin.iOS, Xamarin.Android & Xamarin.Forms.



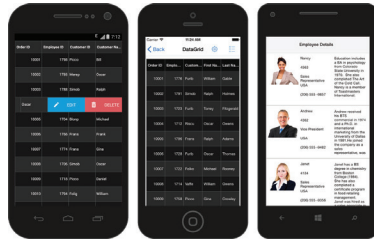
Publisher: Syncfusion | Category: Presentation Layer

100+ Xamarin.iOS, Xamarin.Android, and Xamarin.Forms controls that combine the strengths of advanced data visualization and powerful file-format components from Syncfusion and Xamarin.



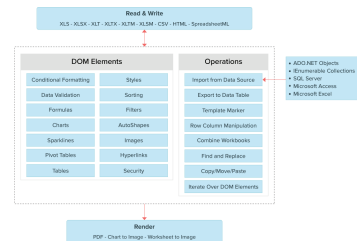
Plot 20+ Chart Types

The Chart control can plot over 20 chart types, from line charts to specialized financial charts. Its rich feature set includes functionalities like data binding, multiple axes, trackball, tooltip and zooming.



Essential DataGrid for Xamarin

A grid component built to achieve the best performance on the Xamarin platform. It offers responsive touch, scrolling 100,000s of records and includes advanced features such as grouping, sorting and filtering.



Read/Write Excel Files

Essential XlsIO is a library that can read and write Microsoft Excel files. It includes a comprehensive API, has no external dependencies and can be used on systems without Excel installed.

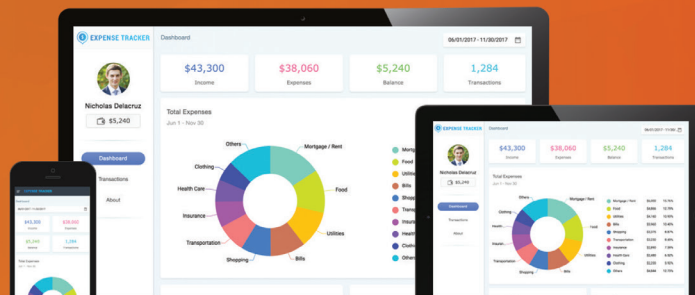


Prices from \$ 975.10
www.componentsource.com/syncfusion-essential-studio-for-xamarin



Syncfusion Essential Studio for JavaScript

Featuring Syncfusion's next-generation pure JavaScript package, Essential JS 2.



WHAT'S NEW?

- New web-based theme studio.
- Comprehensive toolkit of 50+ Vue.js components.

Comprehensive. Modern. Free of external dependencies.



Lightweight



Built for
performance



Customizable with
built-in themes



Modular



Responsive



Usable in multiple
languages

LEARN MORE

www.componentsource.com/essential-studio-for-javascript





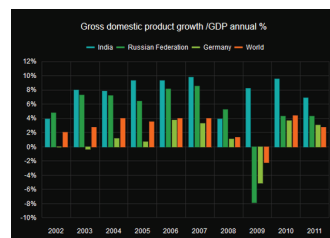
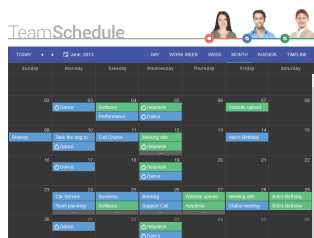
Kendo UI

JavaScript, HTML5 UI widgets for responsive web and data visualization.

Publisher: Telerik | Category: Presentation Layer | ★★★★★

A complete JavaScript UI component library that includes 70+ customizable UI components and allows you to quickly build high quality, high performance responsive web apps using your framework of choice - jQuery, Angular, React or Vue.

Contact Name	Contact Title	Company Name	Country
Giovanni Rovelli	Marketing Manager	Magazzini Alimentari Riuniti	Italy
Dominique Perrier	Marketing Manager	Spécialités du monde	France
Li Naon	Marketing Manager	The Big Cheese	USA
Karin Josephs	Marketing Manager	Toni's Spezialitäten	Germany
Contact Title: Order Administrator			
Christina Berglund	Order Administrator	Berglunds snabbköp	Sweden
Evan Otello	Order Administrator	Drachenstich Delikatessen	Germany



HTML5/JS Grid Widget

Kendo UI's responsive and adaptive HTML5 grid widget offers over 100 features from filtering and sorting data, to advanced features like pagination and hierarchical data grouping.

Powerful HTML5 Scheduler

The Kendo UI Scheduler allows you to easily schedule, display and edit appointments. You can display day, week, month, timeline and agenda views.

Eye-catching JS Charts

The Kendo UI Chart includes all commonly used chart types. Charts are completely rendered through JavaScript, so are server platform agnostic, and boost the performance of your app.



Prices from \$ 881.02
www.componentsource.com/kendo-ui





Modern UI Made Easy



Building a modern UI for Web, Desktop and Mobile apps and Chatbots has never been easier
with our .NET, JavaScript, Productivity and Testing Tools

www.componentsource.com/telerik

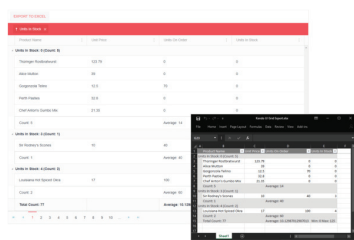


Telerik DevCraft Complete

Quickly build modern, high-performance apps for any platform.

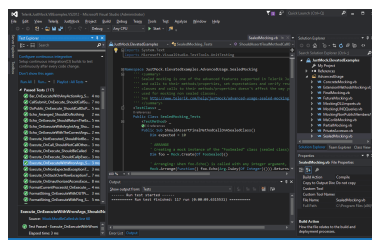
Publisher: Telerik | Category: Presentation Layer

Build .NET and JavaScript apps with a sleek, fast and consistent UI across all web, desktop and mobile platforms. Includes mocking and reporting functionality, and Priority Support with unlimited support tickets and 24 hour response time.



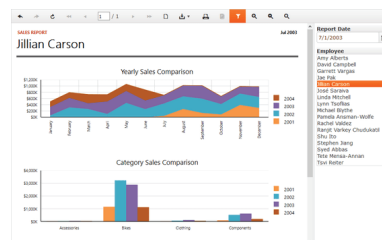
Grid Export to Excel

The Kendo UI grid provides server-agnostic client Excel export functionality. You have the option to customize the rows/columns and cells of the exported file by intercepting the excelExport event.



Flexible Mocking Framework

Telerik JustMock makes it easier for you to create mock objects and set expectations independently of external dependencies like databases, web service calls, or proprietary code.



.NET Reporting Made Easy

Telerik Reporting lets you create, view and export rich, beautiful, interactive and reusable reports – everything a lightweight and feature-complete reporting solution should do.



Prices from \$ 1,469.02
www.componentsource.com/telerik-devcraft-complete



The Ultimate Toolkit for Building Modern Apps with Outstanding UI

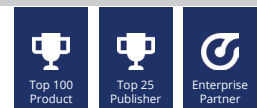
The Ultimate Toolkit for Building Modern Apps with Outstanding UI



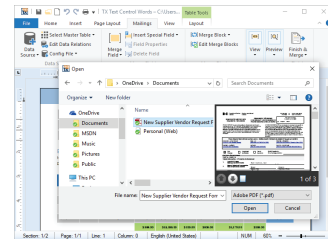
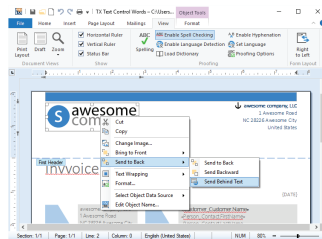
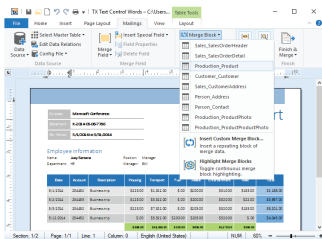
TX Text Control .NET for Windows Forms

Add Microsoft Word look and feel editing to your Windows Forms applications.

Publisher: Text Control | Category: Word Processing



TX Text Control .NET for Windows Forms is a royalty-free, fully programmable rich edit control that offers developers a broad range of word processing features in a reusable component for Visual Studio.



Reporting and Mail Merge

Text Control combines the power of a programmable reporting tool and WYSIWYG word processor. Create mail merge and table reports, master-detail and sub-reports based on MS Word compatible report templates.

Word Compatible File Formats

Modify MS Word documents or create Adobe PDF documents from your application. Supports a range of word processing formats (RTF, DOC, DOCX, HTML, XML, PDF) and image formats (GIF, PNG, JPG, BMP, WMF, EMF, TIF).

Import Adobe PDF Documents

Import PDF documents, and view, edit or convert them. Imported PDFs can be modified just like any other format. The fully featured API can be used to change the content or to search on the document.

TEXTCONTROL

Prices from \$ 1,126.02
www.componentsource.com/tx-text-control-net-professional



TEXTCONTROL

Visual Studio
Partner

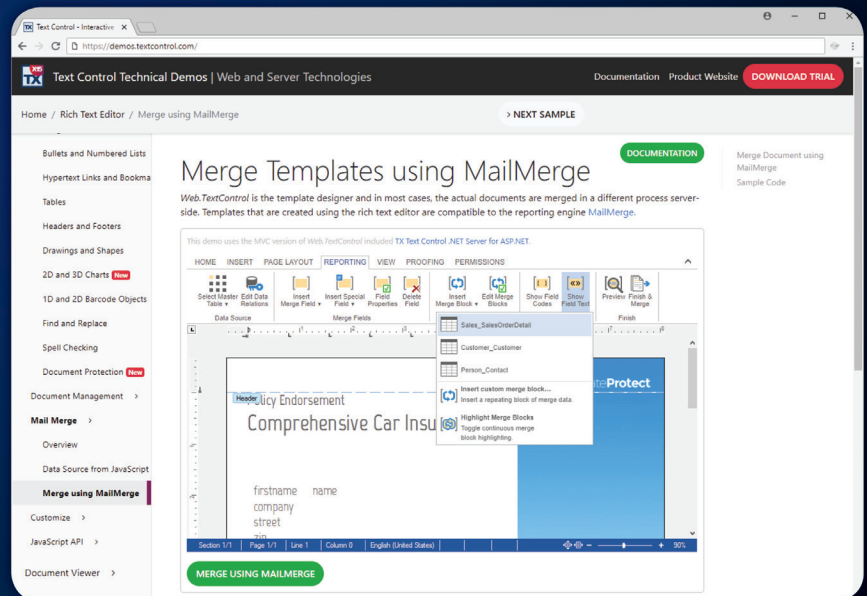
Integrate Documents and Reporting in Any Platform

The Text Control Reporting Framework combines powerful reporting features with an easy-to-use, MS Word compatible, word processor.

Learn more:

componentsource.com/textcontrol

**WE ARE CHANGING
THE WAY YOU LOOK AT
REPORTING**



© 2018 Text Control GmbH. All rights reserved. All other product and brand names are trademarks and/or registered trademarks of their respective owners.

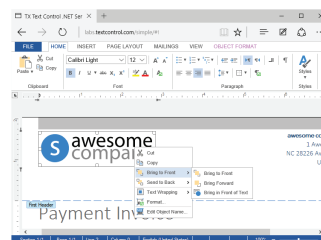
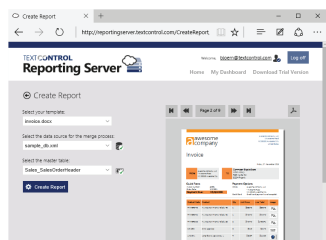
TX Text Control .NET Server for ASP.NET

Powerful word processing & reporting for your Web Forms & MVC Web applications.

Publisher: Text Control | Category: Word Processing



Specifically designed to run in server side applications, TX Text Control .NET Server for ASP.NET is ideal for batch processing or printing large volumes of documents.



Automate MS Word Documents

Replacing MS Office Automation in applications is a typical use case for document APIs such as TX Text Control. Automate, edit and create documents using UI and non-UI components.

Document Conversion

Perform document conversion in batch processes without user interaction. Convert and modify different document types to a standardized style or merge various document types into one unified document.

Cross-Browser Document Editing

Includes a true WYSIWYG, HTML5-based Web editor and reporting template designer. Give your users an MS Word compatible editor to create powerful reporting templates anywhere - in any browser on any device.

TEXTCONTROL

Prices from \$ 2,938.04
www.componentsource.com/tx-text-control-net-server





TEXT**CONTROL**

Create Documents in the Cloud from Any Platform

RESTful Web API powered reporting
platform to create MS Word
compatible reports.



**WE ARE CHANGING
THE WAY YOU LOOK AT
REPORTING**



© 2018 Text Control GmbH. All rights reserved. All other product and brand names are trademarks and/or registered trademarks of their respective owners.

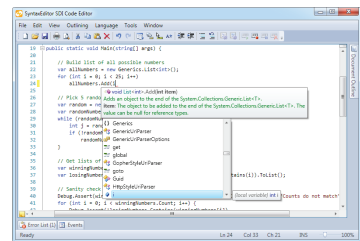
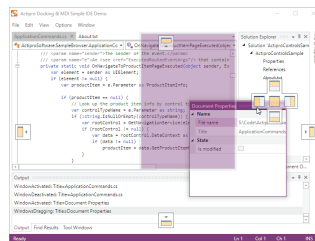
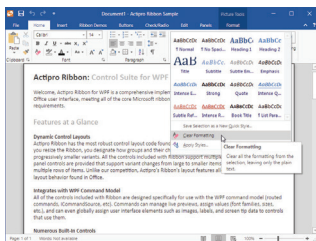
Actipro WPF Studio

Everything you need to add rich functionality to your WPF apps.



Publisher: Actipro Software | Category: Presentation Layer | ★★★★★

Over 100 WPF controls and components including barcodes, charts, datagrid, docking & MDI, editors, gauges, micro charts, navigation, property grid, ribbon, syntax editor, themes, tree controls, views, wizards, and a shared library of controls.



Professional Themes for Apps

Use Actipro's theming system to create a cohesive theme across all the controls in your application, and include Actipro's ribbon control to implement a familiar modern interface for productivity applications.

Docking, MDI & Property Grids

Actipro's docking windows & MDI controls provide the modern features found in major IDEs, including appealing animations. Combine with Actipro's property grid & syntax editor to make distinctive applications.

Code Editing in Your Apps

SyntaxEditor is a powerful text editing control that is packed with features for efficient code editing, including syntax highlighting, code outlining, parsing, line numbers and block selection.



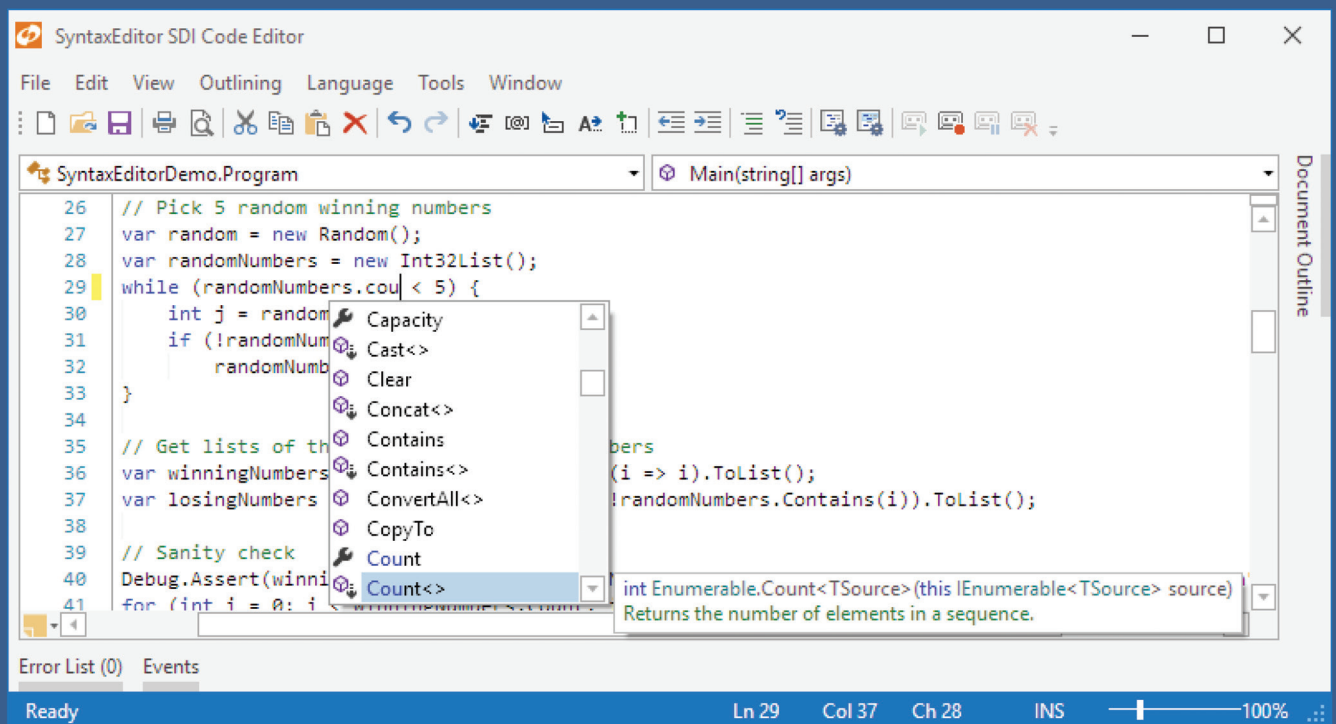
Prices from \$ 636.02
www.componentsource.com/actipro-wpf-studio



Actipro SyntaxEditor

for WPF, Universal Windows, Silverlight, WinForms

*The ultimate syntax-highlighting
code editor control*



Discover the possibilities...
componentsource.com/actipro



Highcharts Suite

Add interactive charts and maps to your apps and Web sites.



Publisher: Highsoft | Category: Data Visualization | ★★★★★

Highcharts Suite provides an easy way to integrate responsive, SVG-based maps and charts with your Web and mobile projects. The charting library allows for great flexibility and easy styling via JavaScript or CSS.



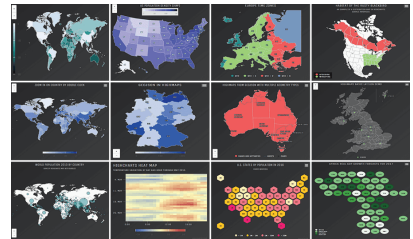
Highcharts

Chart types include line charts, area charts, column & bar charts, pie charts, scatter charts, bubble charts, polar charts, pareto charts, funnel charts, bell curves, heat maps, tree maps and gauges.



Highstock

Create sophisticated dynamic financial charts or general timeline charts that work with any back-end database or server stack. Data can be given in any form, including CSV, JSON or loaded and updated live.



Highmaps

Display geographic data with ease. Highmaps provides map area, map line, points of interest, bubbles on maps, heatmaps, and more. Highmaps intelligently adapts to any device and supports multi-touch gestures.



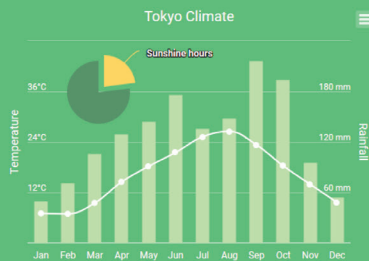
Prices from \$ 1,254.40
www.componentsource.com/highcharts-suite



Make your data come alive

Easily add interactive, responsive charts, stock charts and maps to your website.

Highcharts



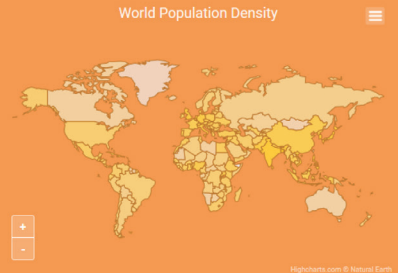
Used by hundreds of thousands of developers and over 80 out of the world's 100 largest companies, Highcharts is the simplest, yet most flexible charting API on the market.

Highstock



Create stock or timeline charts in pure JavaScript. Includes technical indicators and sophisticated navigation options like a small navigator series, preset date ranges and scrolling.

Highmaps



Build interactive maps to display sales, election results or any other information linked to geography. Perfect for standalone use or in dashboards in combination with Highcharts!

Also available: **Highcharts Suite** includes Highcharts, Highstock, Highmaps, Highcharts Editor and Highcharts Export Server.

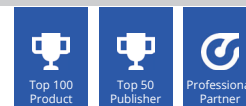


Learn more at www.componentsource.com/highsoft

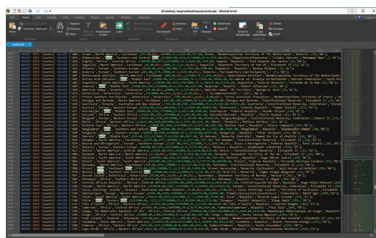
UltraEdit

Provides rich text and source code editing capabilities.

Publisher: IDM Computer Solutions | Category: Coding & Debugging | ★★★★★

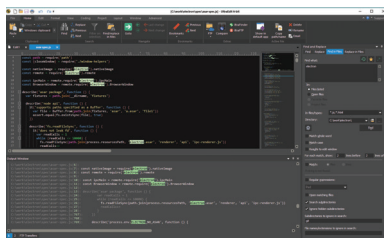


UltraEdit is widely known for its rich and extensible code highlighting, powerful find and replace, integrated FTP, SSH, and file compare capabilities.



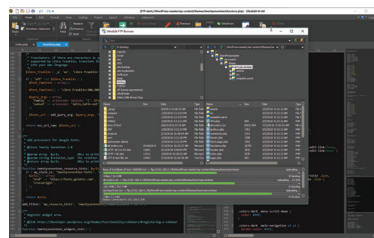
Large File Handling

UltraEdit was created to edit the huge files (4 GB and beyond) that cause other text editors to crash. It is useful for people who deal with databases and large log files.



Find and Replace

From quick inline finds to massive replaces across your entire project. Find/replace across files, Perl regular expressions, column-based search, line filtering, and more.



Includes FTP, SSH, and File Compare Tools

With an integrated FTP browser, SSH/telnet, file compare, scripting, custom tools, macros and smart templates, UltraEdit is more than just a text editor.



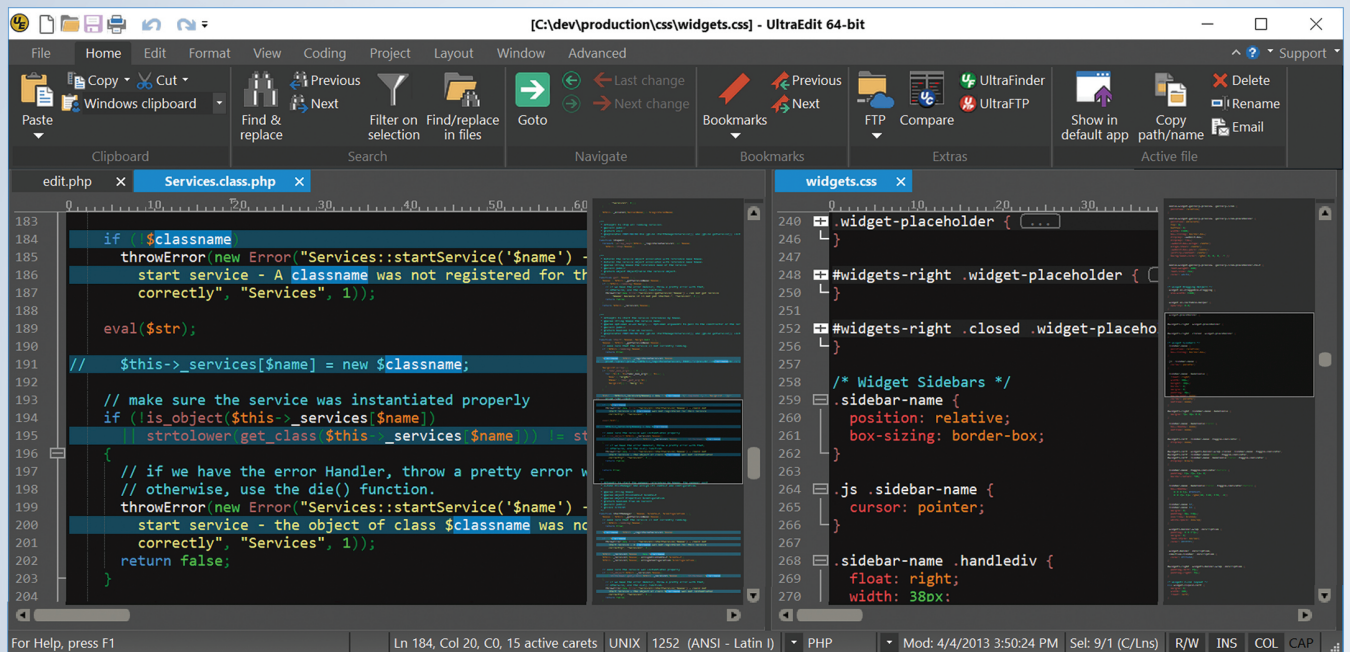
Prices from \$ 97.95
www.componentsource.com/ultraedit





There is no substitute for **proven**.

UltraEdit: The world's #1 text editor.



Proven, reliable, secure, trusted...supported. Used by Fortune 500/1000 companies... trusted by millions. For over 25 years.

Learn more: www.componentsource.com/idm



Microsoft
Partner



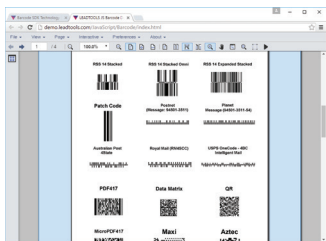
LEADTOOLS Document Imaging Suite SDK

Develop powerful document imaging applications.



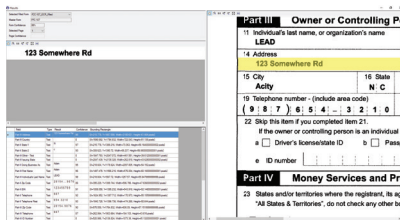
Publisher: LEADTOOLS | Category: Image Editing & Processing | ★★★★★

Build end-to-end document imaging solutions that require OCR, OMR, barcode, forms recognition and processing, PDF, conversion, annotation, HTML5 Zero-footprint viewing, print capture and image viewing functionality.



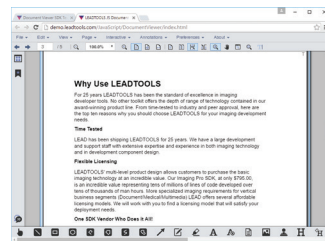
Barcodes

LEADTOOLS supports more than 100 different barcode types and sub-types such as UPC, EAN, Code 128, QR Code, Data Matrix, and PDF417, making it the right choice for any barcode application in any industry.



Forms Recognition

In addition to basic forms recognition with static field locations, LEADTOOLS is able to detect and process unstructured and semi-structured documents such as invoices, driver's licenses, and passports.



Document Viewer

Create robust, fully featured applications with rich-document viewing features, including text search, annotation, memory-efficient paging, inertial scrolling, and vector display.

LEADTOOLS®

Prices from \$ 4,895.10
www.componentsource.com/leadtools-document-imaging-suite



LEADTOOLS®



OCR

FORMS

BARCODE

DICOM

PACS

+ MANY
MORE

One SDK, Multiple Platforms

LEADTOOLS toolkits will help you create desktop, web, server, and mobile applications with the greatest collection of programmer-friendly and cross-platform document, medical, and multimedia technologies.

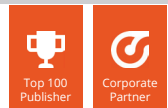
Get Started Today

[COMPONENTSOURCE.COM/LEADTOOLS](https://componentsource.com/leadtools)

**LEAD
TECHNOLOGIES**
INCORPORATED

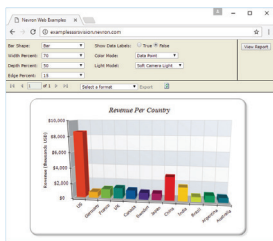
Nevron Vision for SSRS

Advanced data visualization for more informative reports.



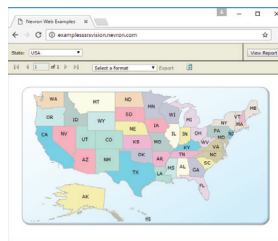
Publisher: Nevron | Category: Data Visualization | ★★★★★

Nevron Vision for SSRS delivers enhancements to your Reporting Services reports. It offers amazing data visualization features, performance and flexibility, directly within the native SSRS environment.



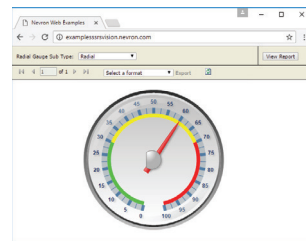
Nevron Chart for SSRS

Delivers a complete set of 2D and 3D chart types, customizable axes, data integration and visual effects to your reports. An indispensable tool for any Business Intelligence, scorecard or scientific report.



Nevron Map for SSRS

Provides report authors with an easy and powerful way to create choropleth and informative maps that are integrated with data. A perfect complement to any report that displays geographical data of any type.



Nevron Gauge for SSRS

Delivers a complete set of gauge types, highly customizable axes, tight data integration and unmatched visual effects to your reports. It includes linear gauges, state indicators and numeric displays.

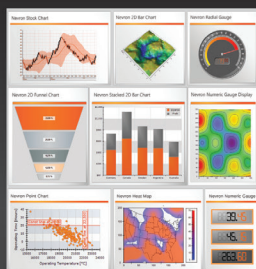


Prices from \$ 1,177.05
www.componentsource.com/nevron-vision-for-ssrs





Nevron provides leading data visualization and UI controls for a diverse range of .NET centric platforms. Our controls and plugins integrate in desktop and web applications: WinForms, WPF, Silverlight, Xamarin, ASP.NET, SSRS, and SharePoint.



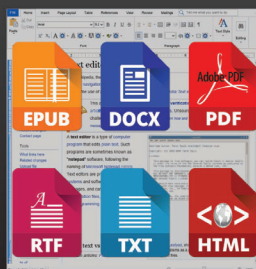
Charts & Gauges

Our leading charting technology allows developers and IT professionals to create presentational, business, scientific, financial, real-time monitoring, statistical, process control, BI, dashboards, and other types of charts. It includes features such as 2D/3D hardware acceleration, binding to a diverse range of data sources, advanced axes, numerous charting types and many others.



Diagrams

Industry-leading diagramming components that can help you author and visualize complex flowcharts, org-charts, mind-maps, ERM, UML and other types of diagrams. Features include automatic layouts, interactive editing, routable connectors, groups, smart shapes, line jumps, Visio import, export to SVG and PDF, and many others.



Text processing

Industry leading control for DOCX, RTF, PDF, HTML and EPUB document generation, editing and visualization. Our control allows you to create all types of text documents such as contracts, invoices, reports, and others. Suitable for advanced text processing, automated document generation, reporting, text document conversion, WYSIWYG text and HTML editing, proofreading, and others.

Grids

High performance grid components for displaying, editing and analyzing tabular and hierarchical data. Features include binding to diverse data sources, grouping, filtering, sorting, master-details and advanced extensibility options.

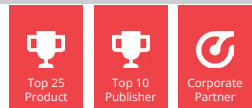
Learn more at www.componentsource.com/nevron today



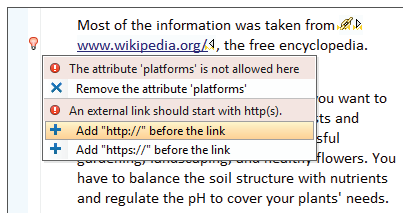
Oxygen XML Editor Enterprise

A complete solution for XML development and authoring.

Publisher: SyncRO Soft | Category: Structured Document Authoring

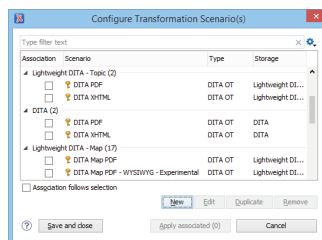


A collection of must have tools for XML editing, with support for all types of XML documents and other file types, including XML Schemas, CSS, XSLT, WSDL, RelaxNG, Schematron, Ant, XQuery, and many more.



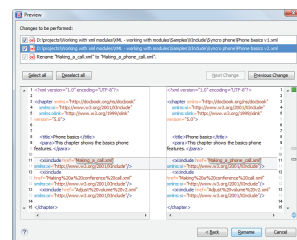
Intelligent XML Editor

XML editing is easier than ever with the help of intelligent actions and easy to use features. Oxygen Quick Fix helps you resolve errors in an XML document by offering quick fixes to problems such as missing required attributes or invalid elements.



Single-Source Publishing

The XML Editor offers preset and configurable scenarios that are one click away, allowing you to produce outputs in PDF, ePUB, HTML, and many other formats using the same source.



Update References

When an XML, XSL, XSD, or WSDL resource is renamed or moved in the Project view, oXygen XML Editor gives you the option to update the references for that resource.



Prices from \$ 945.45
www.componentsource.com/oxygen-xml-editor-enterprise



<oXygen/>® XML Editor



The Complete Solution for
XML Authoring, Development
and Collaboration

schematron
Structured
editing
XML
review
XQuery
Publish
PDF
Collaboration
oXygen
authoring
XML Editor
XSD SCH XSD
XPR RNCFO
frameworks
Profiling
DITA
styles
visual
WebHelp
WSDL
TEI
XSL
PHP
Ant
Js

JS
KML
XSLT
SVN
JSON
SVG
IDREFS
WebDAV
Single
Source
Database
XHTML Read
Change
Collabor
WebDAV

www.componentsource.com/oxygen-xml-editor

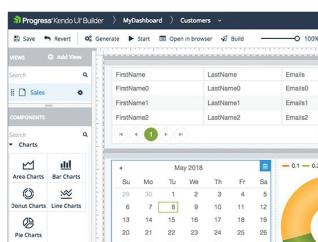


Telerik DevCraft Ultimate

All Telerik .NET tools & Kendo UI JavaScript components in one package.

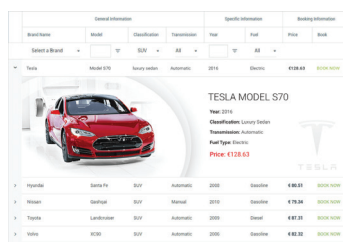
Publisher: Telerik | Category: Presentation Layer

Spans all .NET and JavaScript technologies, delivering outstanding UI with a broad array of styling and customization. Now includes Test Studio Developer Edition, Telerik Report Server, and access to live on-line training.



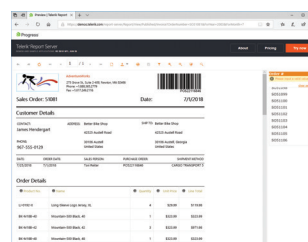
Kendo UI Builder

Create rapid prototypes for quick application scaffolding to jumpstart your Angular or AngularJS development. Increase the productivity of your existing code-based design flow by easily adding complex UI components to your design.



Sleek UI Through Modern Themes

Standardize the look and feel of your apps across the board. Out-of-box themes and limitless customization options allow you to quickly present a professional, well-packaged front-end to your users.



Telerik Report Server

Use Telerik Report Server as a standalone report management solution or integrate it easily with your application. Manage all sever assets through RESTful APIs and create customer business objects as report data.



Prices from \$ 2,155.02
www.componentsource.com/telerik-devcraft-ultimate





Modern UI Made Easy



Building a modern UI for Web, Desktop and Mobile apps and Chatbots has never been easier with our .NET, JavaScript, Productivity and Testing Tools

www.componentsource.com/telerik



Empower your development. Build **better** apps.

GrapeCity's family of products provides **developers, designers, and architects** with the ultimate collection of easy-to-use tools for building **sleek, high-performing, feature-complete** applications. With over 25 years of experience, we understand your needs and offer the industry's best support. **Our team is your team.**



ComponentOne
.NET UI CONTROLS



ActiveReports
REPORTING SOLUTIONS



Spread
SPREADSHEET SOLUTIONS



Wijmo
JAVASCRIPT UI CONTROLS



Documents
DOCUMENT APIs

Learn more at componentsource.com/grapecity

© 2018 GrapeCity, Inc. All rights reserved. All other product and brand names are trademarks and/or registered trademarks of their respective holders.

